תכנון ותכנות

יישומי מחשבים

יצחק עמיהוד M. Sc.

תכנון ותכנות יישומי מחשבים
Design and Programming of Computer Applications

הודפס בישראל
Printed in Israel

מהדורה 1 - דצמבר 1975
הדפסה 2 - מרס 1976
הדפסה 3 - אוגוסט 1978
הדפסה 4 - מרץ 1983

הוצאת "הוד-עמי" ת. ד. 560 רמת-גן

מוקדש להורי

פנחס ומלכה

זכרונם לברכה

הקדמה

גידול משק המחשבים בישראל מעיד על הצורך הגובר באמצעים מתוחכמים לעיבוד נתונים. אנו עדים להרחבת היקף השימוש במחשבים ולהעמקתו: מעיבודים שעיקרם מיון של נתונים ועריכתם להדפסה, לעדכון קבצים ומאגרי נתונים, לשילוב של מערכי יישו-מים למערכות כוללניות אינטגרטיביות והפעלת עיבוד בתקשורת. התקדמות זו הביאה לגידול בדרישה לבעלי מקצוע מתאימים: מתכנתים, מתכננים ומנתחי שיטות.

תכנון ותכנות יישומי מחשבים מיועד ללימוד השלבים הראשונים בעיבוד נתונים אוטומטי: תכנון ותכנות. הוא מציג את עקרונות הנושא וממשיך בתיאור ולימוד מפורט של שפות התכנות המבוקשות כיום בשוק המחשבים הישראלי, המינהלי והמדעי כאחד.

החומר המוצג מכוון להכשיר את הקורא לתכנת עבודות מקובלות בעיבוד נתונים מינהלי ומדעי, כשנקודת המוצא היא שלא נדרשת ידיעה מוקדמת בנושא זה. הספר מיועד להיות לעזר לתלמידים בבתי הספר ומאפשר גם לימוד עצמי.

<u>הספר ותוכנו</u>

בגלל השוני הרב בדרישות הלומדים ובשיטות ההוראה בבתי הספר, ערוך הספר בצורה מודולרית אשר תאפשר גמישות בלימוד החלקים השונים:

חלק ראשון - מבוא לעיבוד נתונים,
חלק שני - אלגוריתמים ותרשימי עיבוד,
חלק שלישי - תכנות פורטרן FORTRAN
חלק רביעי - תכנות קובול COBOL
חלק חמישי - תכנות פ.ל. 1 PL/I

בחלק הראשון מוצגים בקצרה עקרונות מערכת עיבוד נתונים, תאור המחשב ויחידות

קלט/פלט, תהליכי תכנון ותכנות, שפות תכנות ומערכת הפעלה.

בחלק השני מוסבר נושא האלגוריתם ומתוארים שלבי ההכנה של תרשימי זרימה ותרשימים מלבניים. תרשימים אלו הם תאור התהליך הלוגי של התכנון וסיכומו ומהוים חלק מתעוד התוכנית. ההצגה בחלק זה מבוססת בעיקר על פתרון בעיות בתחומי עיבוד שונים.

החלקים הבאים של הספר מוקדשים להוראת תכנות בשלוש השפות העיקריות הנפוצות במערכות מחשבים בינוניים וגדולים: פורטרן (FORTRAN) - ליישומים מדעיים, קובול (COBOL) - ליישומים מינהליים ו-פ.ל.1 (PL/I) - ליישומים מינהליים ומדעיים כאחד. החומר הנלמד מלווה במספר רב של דוגמאות ובפתרונות של בעיות להסבר ולהמחשה.

תכנות בניב כלשהו של שפה מכוון להפיק תוכנית אשר תופעל במחשב מסויים, בפיקוח מערכת הפעלה מוגדרת ותשתמש בשיטות ארגון הקבצים האפשריות במכלול זה. מכיון שכוונת הספר ללמד את יסודות התכנות בניבים שונים של השפות, השתדלתי לשמור על כלליות ההצגה של הנושאים השונים כדי שיתאימו לתחום רחב של מחשבים ומערכות הפעלה. מסיבה זו גם הצגתי את שיטת ארגון הקבצים הסיקוונציאלית (sequential) בלבד כדי להמנע מהרחבת היריעה לתחומי ציוד, מערכות הפעלה ותוכנה של היצרנים השונים.

כדי להפעיל את התוכנית במחשב מסויים צריך הלומד לקבל מידע על נוהלי ההפעלה הנהוגים במתקן שבו הוא עומד להריץ את עבודתו ועל כרטיסי הבקרה (control cards) הנדרשים.

מכיון שמחשבי חברת י.ב.מ. הם הנפוצים ביותר, הוזכרו ציוד ושפות תכנות של חברה זו, וזאת מבלי לפגוע בכלליות ואפשרות ההתאמה גם למערכות מחשבים של יצרנים אחרים.

כל חלק בספר יכול להחשב כיחידת לימוד עצמאית, אולם מומלץ להתחיל בשני החלקים הראשונים ואח"כ לעבור, לפי הצורך, ללימוד שפת התכנות הרצויה. במקרים מסויימים שבהם יתקשה הלומד בפרק כלשהוא בשפת תכנות אחת, מוצע לעיין בהסברים הכלליים לאותו נושא בחלק הדן בשפת תכנות אחרת.

כחומר רקע וכמבוא בנושא עיבוד הנתונים האוטומטי מומלץ הספר "המחשב האלק-טרוני ועיבוד הנתונים" מאת המחבר, אשר מתאר בהרחבה את עקרונות המערכות האו-טומטיות לעיבוד נתונים, את המחשב האלקטרוני ומרכיביו, עקרונות התכנות, שפות תכנות ומערכות הפעלה, התארגנות להתקנה של מערכות יישומי מחשב, ניתוח ותכנון מערכות, הפעלת מערכות מחשבים, ונותן דוגמאות ליישומים בשטחי מינהל ומדע.

באשר למינוח המקצועי בספר, השתדלתי להשתמש בשפה העברית ככל שניתן. במקרים בהם המונח העברי אינו נפוץ או שאינו מוסכם הצגתי גם את המונח באנגלית, ובמקרים מסויימים השתמשתי במונח האנגלי בלבד.

בכתיבת הספר סייעו בידי חברי לעבודה בחברת י.ב.מ. בישראל, והריני מודה להם על כך: יחיאל האן - אלגוריתמים ותרשימי עיבוד; דן פלדמן - תכנות FORTRAN; דני גלבוע - תכנות COBOL ; אלי שלו - תכנות PL/I.

תודתי נתונה לחברת י.ב.מ. על הרשות להעזר בפרסומיה ולהציג את התמונות בחלק ראשון של הספר.

יצחק עמיהוד

רמת-גן, חשון תשל"ו
אוקטובר 1975

תוכן הענינים

עמ'

עמ'

חלק ראשון:

מבוא
לעיבוד נתונים

פרק 1: מערכת אוטומטית לעיבוד נתונים.

מהו עיבוד נתונים

עיבוד נתונים הינו מושג נרחב ביותר המשמש לתאור סדרה של פעולות מתוכננות על נתונים בכדי להשיג תוצאה נדרשת. הגדרה קצרה יותר תהיה: הפיכת נתונים למידע.

הנתונים (Data) הם בטוי לעובדות מסוימות שעשויות להיות בעלות משמעות שמושית למקבל אותן. צרופן של עובדות אלו בצורות שונות וקביעת השייכות ביניהן לבין עובדות קודמות שנאספו, מהווה את פעולת עיבוד-הנתונים (Data-Processing) הנותנות את התוצאה - המידע (Information).

נובע מכך שהמידע הוא הידיעה המשתמעת מתוך הנתונים. מכיון שהמטרה הסופית היא אסוף ידיעות לשם קבלת החלטות בנושאים שונים, הרי שהנתונים חייבים להיות רלוונטיים לאותם נושאים, מדויקים ומוגשים במהירות בזמן הנדרש.

יש לציין שחומר המהוה מידע לדרג מנהלי אחד יכול להיות נתונים לדרג אחר. כדוגמא לכך נקח את התנועות הכספיות בחשבונו של לקוח בבנק המשמשות כנתונים לידיעת יתרת החשבון. זהו המידע עבור הלקוח; עבור מנהל הבנק מהוות כל היתרות נתונים לידיעת מאזן הבנק.

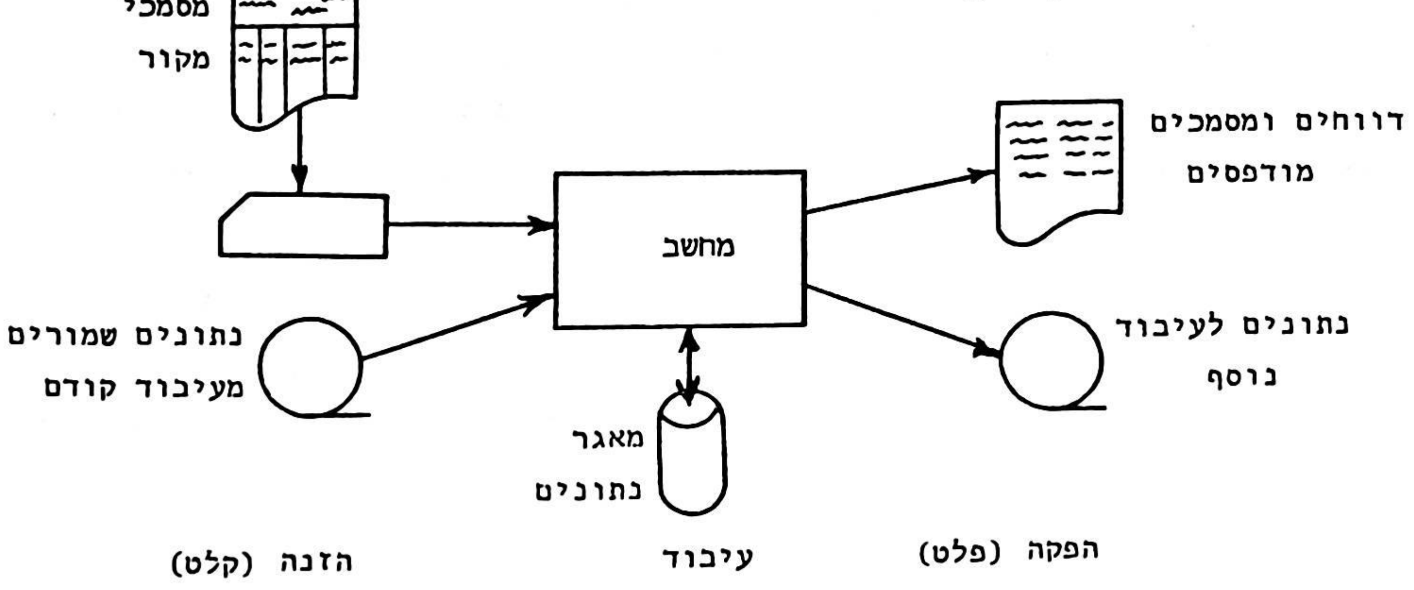
השלבים בעיבוד נתונים

תהליך עיבוד נתונים מורכב משלשה שלבים עיקריים:

א. הזרמת נתונים למערכת העיבוד וקליטתם;

ב. עיבוד נתונים;

ג. הפקת הפלט הדרוש (מידע).



הזרמת נתונים למערכות העיבוד וקליטתם

הנתונים המוזנים למערכת נאספים ממסמכי מקור שונים, כגון: - טופס הוצאה ממחסן, טופס רישום שעות עבודה או טופס בו רושם הרופא את פרטי המחלה של החולה. הזנת הנתונים למערכת מחייבת העברתם ממסמכי המקור לצורת קלט המובנת למערכת העיבוד. אמצעי הקלט הראשוניים הם הכרטיס המנוקב, סרט הנייר המנוקב וכן מס-מכים מודפסים או כתובים במיוחד (בשיטת הקריאה האופטית ובשיטת הקריאה המגנטית).

הנתונים המוזנים למערכת חייבים להיות מהימנים ככל שניתן להשיג. כשה-נתונים מתקבלים ממסמכי המקור עצמם, מידת מהימנותם גבוהה יותר כי כל טיפול בהעברת נתונים ממסמך למסמך בצורה ידנית הוא מקור לשגיאות. מסיבה זו פותחו המתקנים לקליטת מסמכי מקור, כמו קורא המסמכים האופטי.

בנוסף למערכת הבקורת החיצונית של מסמכי המקור לפני המרתם לכרטיסים או סרטים מנוקבים, נערכות בשלב הקלט הבדיקות הבאות:

א. **שלימות** - האם הנתונים נתקבלו במלואם?
דוגמא: בכרטיסים המשמשים לחישוב משכורות עובדים חייבים להיות כל הנתונים לחישוב. אם חסר אחד הנתונים, למשל מספר המחלקה בה עבד עובד מסוים, לא תחושב לו המשכורת.

ב. **סבירות** - התאמת הנתונים למושגים קבועים.
דוגמא: נתון מספרי אסור שיכלול אותיות. תאריך צריך להיות סביר כלומר במסגרת 12 חדשים, 31 יום בחודש.

ג **התאמה** - התאמת נתונים שונים זה לזה מבחינה מהותית.
דוגמא: גליונות תשלומי המשכורות במחלקה במפעל כוללים את פירוט משכורות העובדים וסכום המשכורות הכללי. בעת הקליטה נבדקת ההתאמה הכספית של הסכומים החלקיים (המשכורות) לסכום הכללי.

כמו בכל פעולות בדיקה אי אפשר להבטיח בשלב הקליטה בקורת מושלמת. המשך הבקורת יעשה בשלב עיבוד הנתונים. שלב זה הוא אמנם מאוחר מדי, אבל לעתים אין אפשרות לאתר סוג מסוים של שגיאות אלא בו.

עריכה - השלב האחרון בקליטה הוא עריכת הנתונים ומיונם לפני מסירתם לעיבוד.

עיבוד הנתונים

עיבוד הנתונים כולל פעולות שונות שהן:

א. פעולות אריתמטיות פשוטות ומסובכות;

ב. קבלת החלטה המבוססת על קריטריון כמותי. זהו שלב המסקנה לאחר השואה של תוצאות חישוב כל שהוא אל נתון קבוע;

ג. הסקת מסקנות וחזוי התפתחות עתידה על סמך מכלול נתונים מתקופה קודמת (כאן נכנס אלמנט החישוב הסטטיסטי).

שלב עיבוד נתונים כולל פעולות ביקורת שונות. מגמת פעולות אלו להבטיח ככל האפשר מהימנותם של נתונים שעומדים לעבר, ולפעול בצורה כזו שתדחה את הנתון השגוי או לפחות תאפשר את תיקון השגיאה בשלב פעולות אחר.

הפלט

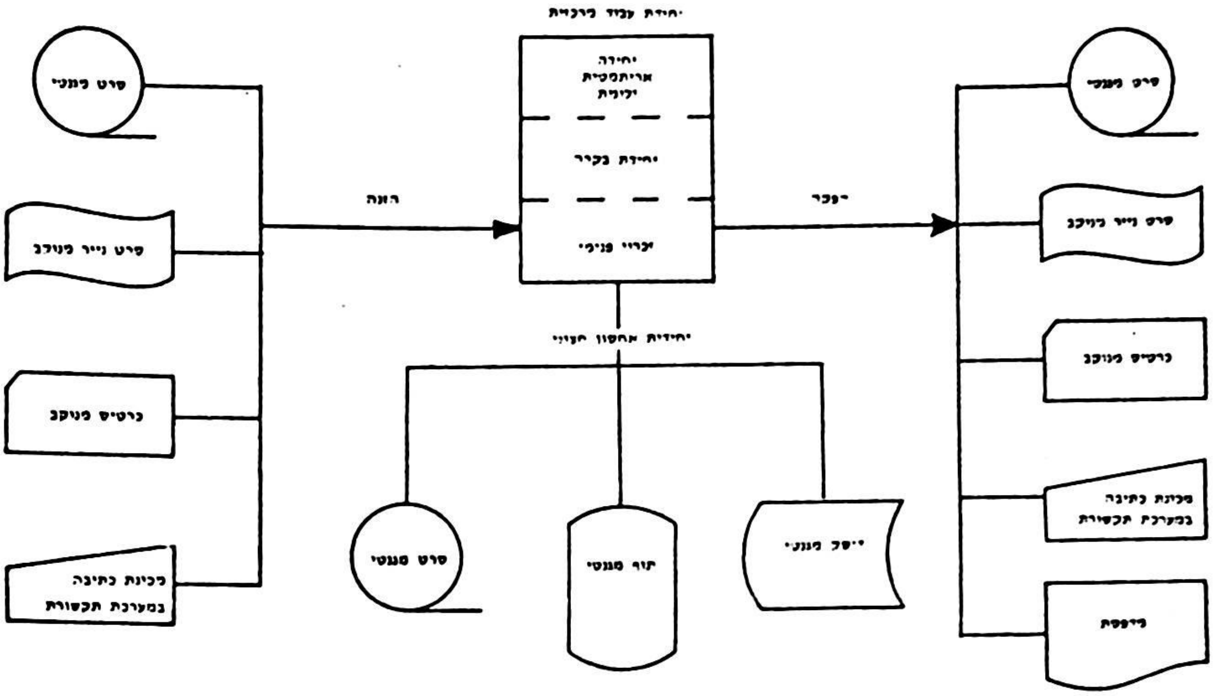
מערכת הפלט מיועדת לספק את התוצאות המתקבלות תוך כדי עיבוד הנתונים: דו"ח, מסמך או ניתוח סטטיסטי כל שהוא. תוצאות אלו יש למסור בצורה הניתנת להבנה ולשימוש על ידי מקבליהן, ולכן נודעת חשיבות מיוחדת לקביעת הפלט הדרוש ולכוון המערכת להשגת מטרה זו. אם כי נראה לכאורה ששלב העיבוד הוא מרכזי בשלבי הפעולות, אין להתעלם מכך שמטרת הקמת המערכת היא קבלת הפלט ולא עיבוד הנתונים, שהוא רק אמצעי להשגת המטרה

המושג "פלט" משמש גם לכנוי התוצאות של תהליכי עיבוד המשמשים לתהליך חוזר במחשב. הפלט מתקבל על אמצעי אחסון מגנטיים או כרטיסים מנוקבים שהמחשב יכול ל"קרוא" בתהליכי עיבוד חוזרים.

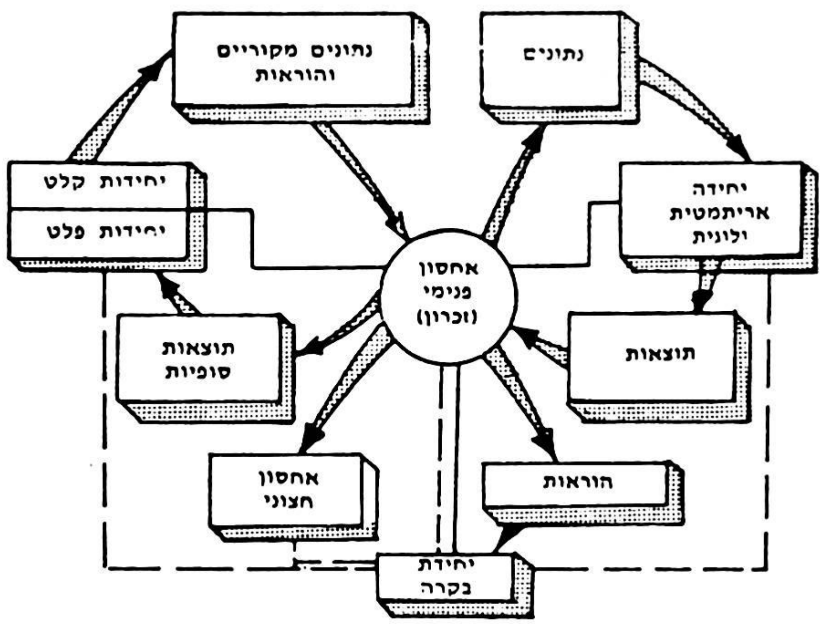
מהי מערכת אוטומטית לעיבוד נתונים

מערכת אוטומטית לעיבוד נתונים משלבת בתוכה את כל הפעולות והשלבים הנדרשים כדי לבצע את המשימה של עיבוד נתונים. המערכת צריכה לענות על הבעיות של עיבוד נתונים ע"י מיכון בעזרה ידנית מינימלית, כאשר המגמה היא להגדיל את היחס בין עבודת מכונה לעבודה ידנית. ברור אם כן, שישנם דרגות שונות של אטומציה במערכות לעיבוד נתונים המבוססות על סוג הציוד העומד במרכז המערכת ומקנה לה את התואר "אוטומטית" קיימת מערכת של מכונות משרדיות להנה"ח ומערכת משוכללת יותר לעיבוד נתונים בשיטת הכרטיס המנוקב. שלב אוטומציה גבוה יותר הושג ע"י הפעלת מערכת עיבוד נתונים שבמרכזה עומד המחשב, כמעבד הנתונים היחיד במערכת.

כל מערכת לעיבוד נתונים, ללא הבדל הסוג והציוד, יכולה לבצע את כל העבודה המוטלת עליה בתנאי שיופעלו נוהלים מתאימים המבטיחים הזרמת נתונים נכונים (קלט) בזמן הדרוש ותקבענה המטרות אותן צריכה לשרת המערכת (הפלט הדרוש). יש



חרשים מערכת לעיבוד נתונים אוטומטי



תרשים פעולות המחשב

לזכור שיש הבדלים מהותיים ביחס לכושר הבצוע וכתוצאה מכך גם לדרישות שאפשר להציב בפני כל מערכת.

בהמשך נתאר את ההרכב של מערכת עיבוד נתונים המבוססת על מחשב אלקטרוני ונעמוד על חשיבות המרכיבים השונים.

מהו מחשב אלקטרוני (Electronic Computer)

המחשב הוא התקן המסוגל לקלוט נתונים, לעבד אותם לפי תכנית מוכתבת ול-ספק את התוצאות הדרושות.

תכונות המחשב

1. כושר לקלוט ולטפל בכמויות חומר גדולות במהירות רבה ולהחזיק אותן בנפח קטן;
2. יכולת לבצע פעולות מתוכננות מראש על אותם נתונים לפי קריטריונים מוג-דרים מראש ע"י השימוש בתכנית המאוחסנת (Stored Program Concept);
3. אפשרות לשנות את ההוראות תוך כדי מהלך הפקודות בהתאם לקריטריונים שונים;
4. יכולת למסור אל מחוץ למערכת את התוצאה הסופית בצורה הדרושה לשימוש בני-אדם (דוחים) או לשימוש נוסף ע"י המחשב עצמו (קבצים מגנטיים).

במחשב פותחה המערכת הלוגית המאפשרת, יחד עם המערכת האריתמטית, לבצע פעולות מורכבות כמעט ללא הגבלה. היחידה הלוגית מעניקה למחשב את אחת מתכו-נותיו החשובות אם לא החשובה שבהן: היא מאפשרת לו ברירה ובחירה בין דרכי פעו-לה שונות, ומקדמת אותו לקראת דמיון למוח האנושי.

אמצעי הקלט והפלט של המחשב שונים ומגוונים, החל ממכונות העובדות בקצב מכני, כגון מדפסת, קוראת ומנקבת כרטיסים, וכלה בסרטים ודיסקים ומערכות אח-רות שהקריאה והכתיבה בהם היא מגנטית. אמצעי הקלט והפלט נמצאים במרוץ כדי להדביק את התפתחות המערכת המרכזית של המחשב, מבחינת מהירות העברת הנתונים ומהימנותם.

יצוג נתונים במחשב

המחשב הינו מכונה משוכללת ומיוחדת אבל לא יותר ממכונה. המיוחד שבה הוא האפשרות להפעיל אותה בצורה כזו שנדמה יהיה לנו שלפנינו מוח אנושי, ואמנם תכופות נקרא המחשב בשם "מוח אלקטרוני". הסבות שהניעו לכנותו בשם זה הן: כו-שר הקליטה של נתונים, יכולת השמירה (אחסון) שלהם, כושר העבודה (אריתמטי ולוגי)

וכושר ההבעה (הדפסה למשל). במסגרת זו נדון בשיטת היצוג של נתונים בזכרון המחשב ובמתקני האחסון החיצוניים שלו, ונראה באיזו דרך הוא מבצע את הפעולות האריתמטיות והלוגיות המקנות לו את עוצמתו.

אנו משתמשים בשפתנו להבעת מושגים במלים הידועות רק לנו, המשתמשים באותה שפה ולא לאחרים ששפתנו אינה ידועה להם. כך גם משפחות המחשבים "משתמשות" באו-צר סמלים וקודים מוסכמים לקשר בינן לבין האדם. לכל משפחה שיטת יצוג נתונים המיוחדת לה, ובה היא משתמשת לשמירתם במתקן אחסון חיצוני לשמוש נוסף. כדי לה-ביא את נושא עיבוד הנתונים למישור של שיתוף פעולה, קיימת אפשרות להעביר את הנתונים לשיטת יצוג כזו שתתקבל על-ידי כל משפחות המחשבים: הכרטיס המנוקב הוא דוגמה טובה לכך, ובמדה פחותה יותר הסרט המגנטי.

שיטות ספירה במחשבים

אחסון נתונים במחשב וחישובים המתבצעים בו יכולים להעשות בכמה שיטות ספירה:

א. השיטה העשרונית המקובלת בידנו כשיטת ספירה (DECIMAL).

ב. השיטה הבינרית (BINARY).

בשיטה העשרונית מיצגים כל מספר ע"י סדרה של ספרות מ-0 עד 9. בשיטה הבינרית קיימות שתי ספרות בלבד: 0 ו-1. הסיפרה הבינרית נקראת סבית (BIT-Binary Digit). כל סיפרה בינרית במספר מייצגת חזקה של 2 וערכי הפו-זיציות במספר (מימין לשמאל) אינם נמדדים בכפולות של 10, אלא בכפולות של 2, כלומר 16,8,4,2,1 וכו'.

בשיטה הבינרית מספר הספרות רב ואיננו נוח לשמוש ולכן מתארים את המספר הבינרי בשיטה אוקטלית (ספרות 0 עד 7) או שיטה הקסדצימלית (ספרות 0 ... 9, F,E,D,C,B,A) שאינן אלא חזקות של 2 ($2^3 = 8$; $2^4 = 16$).

מן הראוי לציין שבחיי יומיום אנו נתקלים בשיטות ספירה נוספות כגון שיטת הספירה העברית והלטינית שבה ערכים מספריים מוקנים לאותיות ה-א"ב. שיטת ספי-רה אחרת היא זו המקובלת בשעון (כפולות של 12).

לדוגמה נציג את טבלת המספרים העשרוניים וערכיהם בשיטות השונות.

ליצוג ספרות אוקטליות דרושות 3 ספרות בינריות, ובהתאמה - לייצוג ספרות הקסדצימליות דרושות 4 ספרות בינריות כפי שמתואר בטבלה.

ספרות דצימליות מיוצגות גם הן ע"י 4 ספרות בינריות, אם כי אינן מנצלות את כל התחום (הערכים מ-10 עד 15).

עשרונית DECIMAL	בינרית BINARY	אוקטלית OCTAL	הקסדצימלית HEXADECIMAL
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14

האריתמטיקה בשיטות ספירה שונות

1. חיבור

חיבור מספרים בשיטה הבינרית זהה לחיבור בשיטה העשרונית. כמובן שיש לקחת בחשבון ש-1 + 1 = 0 ועוד נשוא של 1, או: 1 + 1 + 1 = 1 ונשוא של 1 וכד'.

דוגמה לחיבור מספרים בשיטות השונות:

בשיטה הבינרית	בשיטה העשרונית
101011 (43)	55
+ 110111 (55)	+ 43
1100010 (98)	98

2. כפל

בחיבור לא רואים בבירור את יתרון השיטה, אולם לוח הכפל הבינרי פשוט לאין ערוך לעומת זה של השיטה העשרונית, וכולל 4 אפשרויות בלבד: 0 × 0 = 0, 0 × 1 = 0, 1 × 0 = 0, 1 × 1 = 1.

דוגמה לכפל בשיטות השונות:

בשיטה הבינרית

```
 1010110   (86)
×    101   (5)
 ---------------
  1010110
 0000000
1010110
 ---------------
110101110 (430)
```

בשיטה העשרונית

```
  86
×  5
 ----
 430
```

ע"י הוספת 0 בצד ימין של מספר עשרוני אנו כופלים אותו ב-10 והוספת 0 במספר בינרי כופלת אותו ב-2.

יצוג תוים, מלים והוראות במחשב

יצוג מספרים בשיטות ספירה

יצוג מספרים בשיטה העשרונית, המקובלת במחשבים שונים, הוא למעשה יצוג הקסדצימלי בו כל ספרה מיוצגת ע"י 4 ספרות בינריות בשיטת BCD (Binary Coded Decimal). ההבדל בין שיטה זו לשיטה הבינרית הטהורה הוא בכך שבשיטה הבינרית מבחין המחשב בכל סבית בפני עצמה בעוד שבשיטה העשרונית הוא קולט את הקבוצה של 4 סביות כסיפרה אחת.

לדוגמה:

המספר 869 מיוצג בשיטה העשרונית במחשב ע"י הצירוף הבא:

1000 / 0110 / 1001

בשיטה הבינרית יראה המספר כך: 1100111011

מהדוגמה הזו אנו רואים שהמספר הבינרי דורש פחות ספרות בינריות ככל שהוא גדול יותר. זהו יתרון נוסף ליתרון החשוב של פשטות המעגלים במחשב לבצוע הפעולות האריתמטיות.

את המספר הבינרי שתארנו קודם, אפשר לרשום ביצוג הקסדצימלי בצורה הבאה:

הקסדצימלי 33B - 1011 / 0011 / 11 (חלוקה לארבע סביות)

אותיות וסימנים במחשב

ברוב המחשבים מיוצגים התוים בקבוצות של 8 סביות ועל-ידי כך אפשר לתאר 256 צרופים שונים.

בדומה לכרטיסים מנוקבים, בהם יש נקוב ספרות ונקוב עילי (שורות V,X,O) כך גם כאן מחולקות סביות התו לשני אזורים: 4 סביות לתאור ערך מספרי ו-4 סביות לתאור קבוצת השיוך של התוים.

ספרה תיוצג ע"י ערכה ההקסדצימלי וארבע סביות השיוך 1111
התוים A עד I יתוארו ע"י ערך מספרי 1 עד 9 וסביות השיוך 1100;
התוים J עד R יתוארו ע"י ערך מספרי 1 עד 9 וסביות השיוך 1101;
התוים S עד Z יתוארו ע"י ערך מספרי 2 עד 9 וסביות השיוך 1110;
שאר האפשרויות שלא באו לידי בטוי בטבלה זו מתארים תוים שונים כמו * ; : ! - $ וכו'. לאפשרויות יצוג רבות אין תאור גרפי מוגדר.

כדי לחסוך באחסון לתאור הספרות המסתפקות ב-4 סביות בלבד, הנהיגו במחשבים הפועלים בשיטה העשרונית דרכים לחסכון ע"י אחסון שתי ספרות בתו אחד.
שיטת יצוג התוים שתוארה מתיחסת לאחסון הנתונים בזכרון הפנימי של המחשב ובאמצעי האחסון החיצוני: סרטים, דיסקים ומתקנים דומים אחרים.

המלה של המחשב

הנתונים השמורים בזכרון המחשב נמצאים בקבוצות של סביות (BITS), הנקראות "מלת מחשב" (COMPUTER WORD). קיימים מחשבים בהם הקבוצה או יחידת אינפורמציה זו קבועה וקיימים אחרים בהם קיימת גמישות מסויימת. מלת המחשב הקטנה ביותר המקובלת היא זו שמספר הסביות בה יכולה ליצג תו אחד לפחות. במחשבים אחרים מלת המחשב גדולה הרבה יותר ויכולה להכיל יותר תוים.

מלה בת תו אחד
8 סביות

תו אלפנומרי אחד

1	2	3	4	5	6	7	8

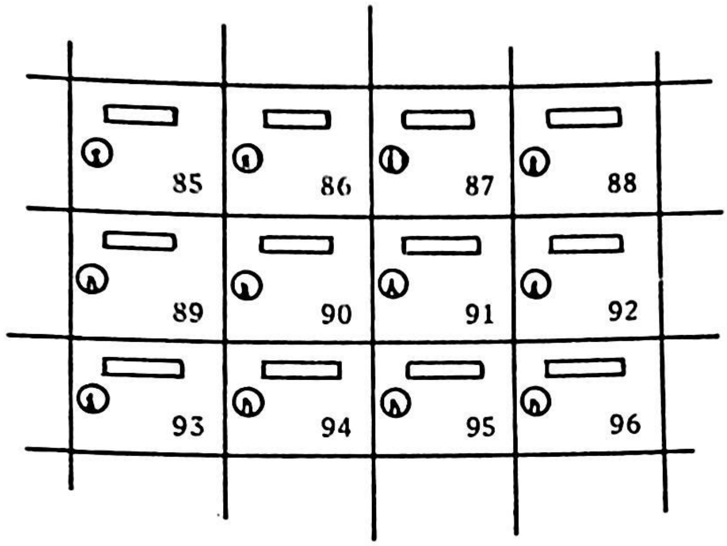
מלה בת 32
סביות

מלה בינרית

1	2	3	4	5__________29	30	31	32

הפניה אל הנתונים הנמצאים בזכרון נעשית על-ידי שמוש בשיטת הכתובות. לכל קבוצת סביות הקטנה ביותר שניתן לפנות אליה במחשב יש כתובת. הכתובת הינה המספר הסידורי של הקבוצה בזכרון המחשב: הראשונה היא 1, השניה - 2, וכן הלאה. גודל הזכרון הפנימי של המחשב מוגדר ע"י הכתובת של ה"מלה" האחרונה בו.

את זכרון המחשב והנתונים המאוחסנים בו ניתן לדמות לארון תיבות המכתבים אשר בבית הדואר. לכל תיבה יש כתובת ומפתח מיוחד, רק לבעל המפתח יש גישה אל התיבה, לשם מילויה או לשם הרקת תוכנה.



ההוראה של המחשב

המחשב מופעל ע"י סדרת הוראות הקרויה תכנית. ההוראות הנן מרכיבים ראשו-ניים של פעולה שאפשר לסדרן בצירופים ובסדר רצוי על מנת להפעיל את המחשב לבצוע העיבוד. הן חייבות להיות חד-משמעיות ובעלות ערכים כמותיים ולא איכותיים.

שיטות לרישום נתונים

זכרון המחשב מוגבל בתכולתו לעומת כמות הנתונים המיועדת לעיבוד, ועל כן הם נמצאים במתקני האחסון החיצוני. כדי להביא את הנתונים אל הזכרון יש להזין אותם ולפלוט אותם שוב לאחר העיבוד. קיימים אמצעים רבים לאחסנת נתונים, מהם המיועדים להעברת נתונים חדשים אל המחשב ומהם המיועדים לקלוט נתונים שעובדו במחשב בשלב אחד ולהחזירם אליו לשלב עיבוד שני.

המחשב יכול לקלוט נתונים מכמה צורות אחסון: כרטיס מנוקב, סרט נייר מנו-קב, סרט מגנטי, מתקני אחסון בגישה ישירה (דיסק מגנטי), טופס מודפס בדיו מג-נסית וטופס לקריאה אופטית.

שיטת רישום הנתונים שונה בכל אחת מצורות האחסון האלה, אולם מתקני הקלט של המחשב הופכים את קוד הרישום לקוד אחיד המקובל באותו מחשב. הנתונים המוד-פסים בדיו מגנטית ובאותיות לקריאה אופטית ניתנים לקריאה ע"י האדם. במדה פחו-תה יותר ניתן לקרוא נקובים או פענוח על הכרטיס. בשאר צורות האחסון ובמיוחד בסרט המגנטי ומתקני גישה ישירה - המחשב בלבד יכול לקרוא את הנתונים.

בפרק הזה נעמוד על שיטת רישום הנתונים בכל אחת מצורות האחסון שהוזכרו.

הכרטיס המנוקב

בשלב ההזנה הראשוני גדולה חשיבותו של הכרטיס המנוקב, ה"גשר" בין לשון האדם ולשון המחשב.

הכרטיס המנוקב הוא אחד מאמצעי הקלט החשובים ביותר. הוא נושא הקלט הרא-שוני שאותו יכול מתקן קלט של מחשב ל"קרוא" ו"להבין". הוא משמש כנושא קלט ישיר למערכת עיבוד נתונים כאשר הוא מוזן אליה, או כנושא קלט עקיף, כאשר נתו-ניו מועברים לאמצעי אחסון אחרים ומהירים יותר (סרט מגנטי, דיסק וכד') ומהם מוזנים למחשב.

הכרטיס המנוקב משמש לכמה מטרות:

א. אמצעי הזנת נתונים למחשב;

ב. אמצעי לאחסון נתונים;

ג. אמצעי להעברת נתונים ממכונה למכונה (הסיבה: קוד הנקוב בכרטיס אחיד לכל המשתמשים);

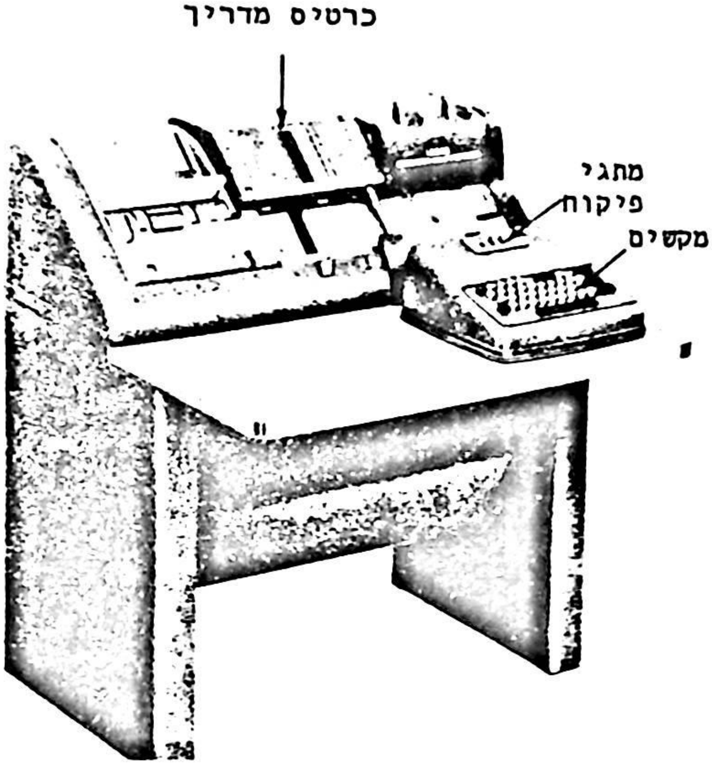
ד. אמצעי להזנתן של תכניות למחשב.

מקובל לקרוא לכרטיס בשם UNIT RECORD (יחידת רישום) ואמנם כזה הוא. בכל כרטיס מנקבים קבוצת נתונים המהוה יחידה לוגית אחת או לעתים שתי קבוצות, אולם נמנעים מבנית יחידות לוגיות הגולשות מכרטיס לכרטיס. כאשר עושים זאת מנקבים בכל הכרטיסים המהוים יחידה לוגית אחת מספר זהוי אחיד. הכרטיסים "נקראים" ביחי דת קלט - קורא כרטיסים CARD-READER. כאשר הפלט הנדרש בעבודה הוא כרטיסים מנו-קבים מבוצעת הפעולה במנקב כרטיסים - CARD PUNCH.

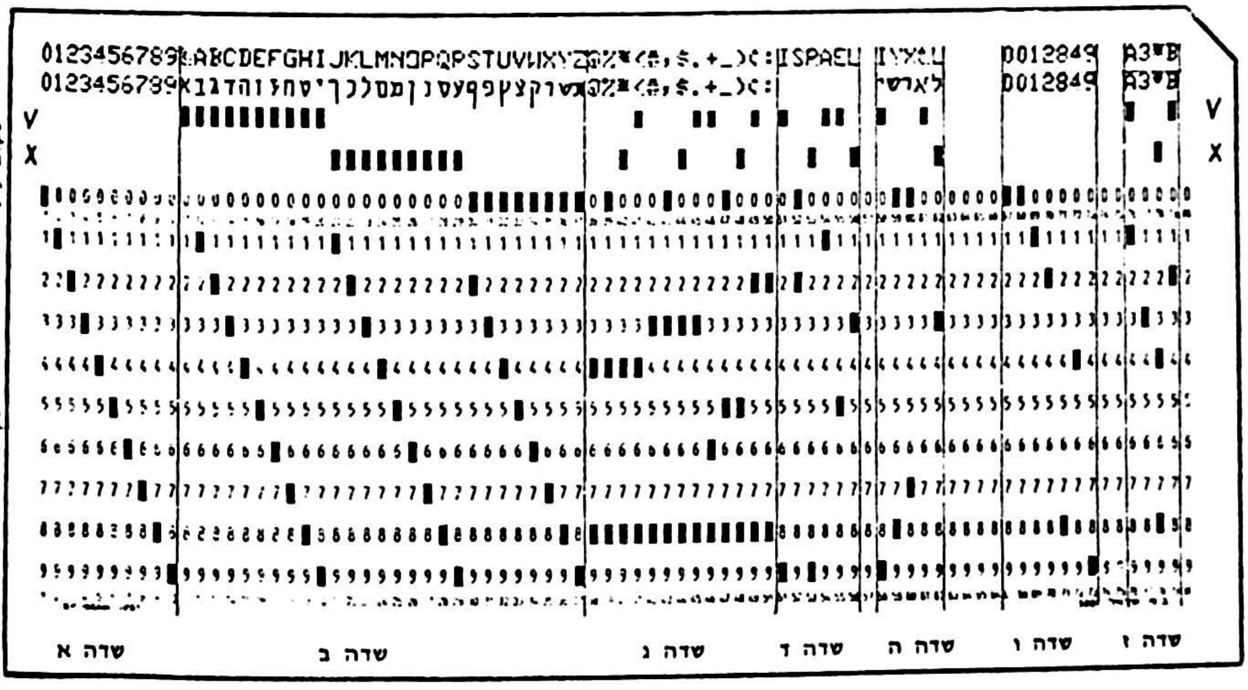
מבנה הכרטיס המנוקב

הכרטיס המוכר הוא בן 80 טורים, אולם נמצאים בשמוש גם כרטיסים בעלי מספר טורים אחר. הכרטיס המקובל מכיל עד שמונים תוים (ספרות, אותיות וסימנים מיו-חדים) המנוקבים בו בזה אחר זה, משמאל לימין לפי קוד קבוע. מספר טורים שבהם מנוקב נתון אחד (סכום, שם, מספר קטלוגי וכד') נקראים "שדה". אוסף שדות המה-וים קבוצת מידע מוגדרת כמו, למשל, מספר קטלוגי, שם פריט, מחיר וכמות במלאי - נקראים "יחידת רישום לוגית" או "רשומה".

הכרטיס מכיל 80 טורים, כאשר בכל טור אפשר לנקב תו אחד בלבד. טורי הכר-טיס מסומנים משמאל לימין בשורה התחתונה. קוד הנקוב כולל אחד, שנים או שלושה נקובים בטור כדי לתאר תו נתון. כאשר אין נקוב בטור מסוים, מבטא אותו הטור את התו "ריק" (Blank). כוון הנקוב משמאל לימין בהתאם לכוון התנועה של הכרטיס במכונת הנקוב (מימין לשמאל).



מנקבת כרטיסים



דוגמה לכרטיס מנוקב (ראה הסבר)

בכל טור בכרטיס אפשר לנקב 12 נקובים שונים. 10 נקובים מסומנים במספרים 0 עד 9. בראש הכרטיס יש מקום לשני נקובים נוספים, הנקראים V,X או נקוב 11 ונקוב 12 (X ליד ה-0 ונקוב V מעליו).

הטורים שנוקבו מפוענחים אוטומטית ע"י המנקבת וערכם מודפס בראש הכרטיס. הפענוח באנגלית או עברית, האותיות העבריות ניתנו בדוגמה רק לשם הסבר.

שיטת הנקוב המקובלת בכרטיסים היא שיטת BCD שבה מקסימום האפשרויות הוא 64 ומספר הנקובים בטור הוא 1, 2 או 3. במחשבי 370 של חברת י.ב.מ. הורחבה מסגרת אפשרויות הנקוב בכרטיס והונהגה שיטת EBCDIC שבה ניתן לנקב בטור אחד מס-פר נקובים, מאחד עד ששה. כל טור מיצג תו אחד או ערך בינרי של תו אחד בן 8 סביות.

תאור השדות (ראה דוגמה מצורפת):

שדה א: טורים 1 עד 10.

בשדה זה נוקבו הספרות מ-0 עד 9, בכל טור נקוב אחד בשורה המתאימה לערך הספרה.

שדה ב: טורים 11 עד 37.

בשדה זה ניתנות במקביל האותיות הלטיניות לעומת האותיות העבריות. האות העברית א חורגת מן הכלל של שני נקובים והיא מקבילה לנקוב של הסימן & באנגלית (נקוב אחד בלבד).

שדה ג: טורים 38 עד 50.

בשדה זה נוקבו סימנים אחדים מן המקובלים כגון: + * או %.

שדה ד: טורים 51 עד 56.

בשדה זה נוקבה המלה ISRAEL. כוון הנקוב משמאל לימין מתאים לכוון הכתי-בה הלטינית.

שדה ה: טורים 58 עד 62.

בין השדה הקודם לשדה זה ישנו טור אחד ריק. כאשר תוכן הכרטיס יוזן למח-שב יכנס למקום המיועד לטור 57 הערך המיצג את התו "ריק". כאשר תוכן הכר-טיס יודפס מן המחשב במדפסת, טור 57 בשורת ההדפסה יהיה ריק. בטורים 58 עד 62 מנוקבת המלה ישראל, משמאל לימין כמובן (לארשי). הפענוח בראש הכר-טיס הוא באותיות האנגליות המקבילות.

כאשר יש להדפיס תוכן שדה אב' עברי יש להפוך במחשב את סדר האותיות על ידי הוראה מיוחדת (Inverse), או ע"י סדרת הוראות כאשר אין הוראה מיוחדת לב-צוע ההפיכה הזו. לכן המלה לארשי בכרטיס תודפס כ-ישראל במדפסת.

שדה ו: טורים 67 עד 73.

בשדה זה בן 7 טורים מנוקב מספר בן 5 ספרות. יש לנקב בצד שמאל של המספר שני אפסים כדי להציבו במקום הנכון בשדה. סיפרת היחידות או הנקודה העשרונית נקבעים לפי הגבול הימני של השדה.

שדה ז: טורים 76 עד 79.

שדה זה מורכב מ-4 תוים שונים: שתי אותיות, סיפרה וסימן *.

קריאת נתונים מכרטיס הנע במתקן נעשית ע"י מברשת מתכת שנמצאת מעל הכרטיס. הפולס החשמלי נוצר ממגע המברשת בגליל הנמצא מתחת לכרטיס בכל פעם שיש נקוב, ומועבר הלאה לעיבוד. ישנם מתקנים לקריאת כרטיסים עם מנגנוני קריאה פוטואלקטריים: בהם קרן אור (ולא מברשת) מזהה את הנקוב. המתקנים המשוכללים לקריאת כרטיסים במחשבים עובדים במהירויות שונות, מ-300 עד 2000 כרטיסים בדקה, בהתאם לדרישות המערכת וסוג הציוד.

כדי לתת מושג על הנפח שתופסים כרטיסים נציין את מידותיהם: גודל הכרטיס בערך 19 × 8,5 ס"מ; עובי של ס"מ אחד יתקבל ע"י 55 כרטיסים; קופסה המכילה 2000 כרטיסים ארכה (ברוטו) כ-40 ס"מ, ותכולתה בתוים - 160,000. לשם השוואה נציג את הסרט המגנטי שרחבו ½ אינטש וצפיפות ההקלטה עליו 1600 תוים לאינטש, כלומר, תכולה של 20 כרטיסים מלאים על סרט באורך של אינטש אחד. קופסה של 2000 כרטיסים נשמרת על סרט מגנטי באורך של 100 אינטש (2.5 מ' סרט).

מנקבת כרטיסים

כשמכונה מדפיסה נתונים היא כותבת אותם בצורה שעין אדם רגילה לקרוא. כאשר המכונה צריכה לרשום אינפורמציה אשר תנתן לקריאה ע"י מכונות אוטומטיות לעיבוד נתונים יש לנקב אותה.

קיימים סוגים שונים של מכונות נקוב. המכונות הנפוצות היום הן בעלות לוח מקשים עברי או אנגלי ומקשים לסימנים מיוחדים, הזנה אוטומטית של כרטיסים ואפשרות העתקה מכרטיס אחד לכרטיס הבא אחריו. המכונות המשוכללות יותר יכולות לפענח את הנקוב ולהדפיסו על ראש הכרטיס.

כוון הנקוב הוא ככוון כתיבת מספרים וכתיבה בלועזית. אי לכך, אינפורמציה א"ב מופיעה בצורה הפוכה, משמאל לימין במקום מימין לשמאל, ובשעת הדפסה יש לדאוג להפיכת כוון שדה נתונים ע"י הוראות מיוחדות למכונת ההדפסה.

תפוקת נקבנית במכונת הניקוב כ-200 כרטיסים לשעה.

סרט מגנטי

במערכת מחשב אלקטרוני לעיבוד נתונים משרת הסרט המגנטי בכמה משימות. אפשר לומר שהמשימה העיקרית והיחידה היא אחסון נתונים וההבדלים מתיחסים למטרות השונות של האחסון.

מהימנות רישום הנתונים וקריאתם, גמישות הרישום וכושר העברה גבוה של נתונים מחד, נוחיות תפעול ומחיר נמוך יחסית מאידך, הביאו לידי כך שהם מקובלים כאמצעי אחסון בכל מערכות המחשבים, גם באלה המבוססות על אמצעי אחסון בגישה ישירה.

משימות האחסון הן:

א. **אחסון נתונים קבוע:** בתפקידו זה ממלא הסרט המגנטי את מקומם של הכרטסות למיניהן המקובלות בעבודה ידנית, במכונות להנהלת-חשבונות או במערכת מכונות לכרטיסי נקוב.

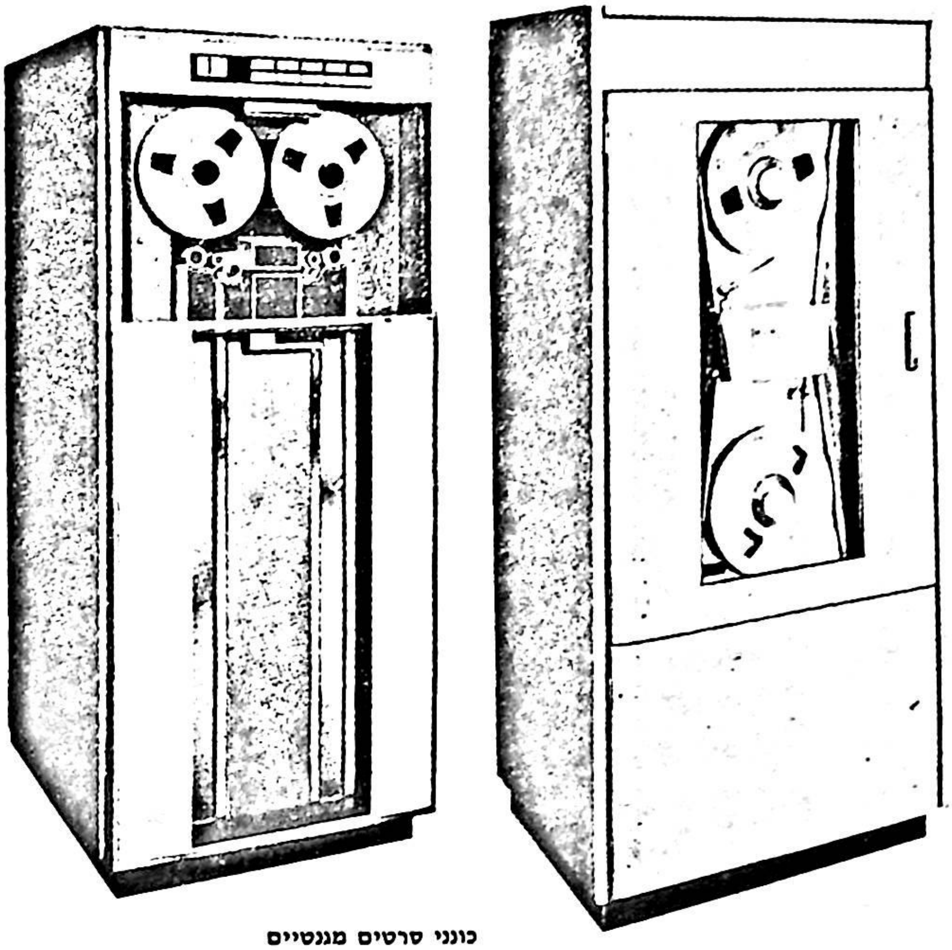
ב. **העברת נתונים בין תהליכי עיבוד:** למטרה זו משמש הסרט המגנטי כאמצעי הפקה והזנה של נתונים. העברת הנתונים אינה חייבת להתבצע מיד והסרט משמש על כן כאמצעי אחסון של נתונים לפרק זמן קצר או ארוך בהתאם לצורך.

ג. **הזנת של נתונים חדשים למערכת:** במחשבים בינוניים וגדולים התגברו על איטיות פעולתם של אמצעי ההזנה האלקטרו-מכניים (קוראי כרטיסים מנוקבים, קורא סרט נייר מנוקב, קוראי מסמכים מגנטיים ואופטיים) ע"י העברת הנתונים ממתקנים אלו אל סרט מגנטי. העברה זו נעשית במתקן מיוחד, במחשב קטן, או בעבודה מקבילה במחשב גדול בשיטת Multiprogramming והסרט המגנטי הנוצר מוזן לתהליכי הקלט של הנתונים. כדי לחסוך את העברת הנתונים מכרטיסים ומסרט נייר אל סרט מגנטי פותחו מתקנים לכתיבת נתונים באופן ישיר על סרט מגנטי שבהם נעשית פעולה זהה לפעולת הנקוב.

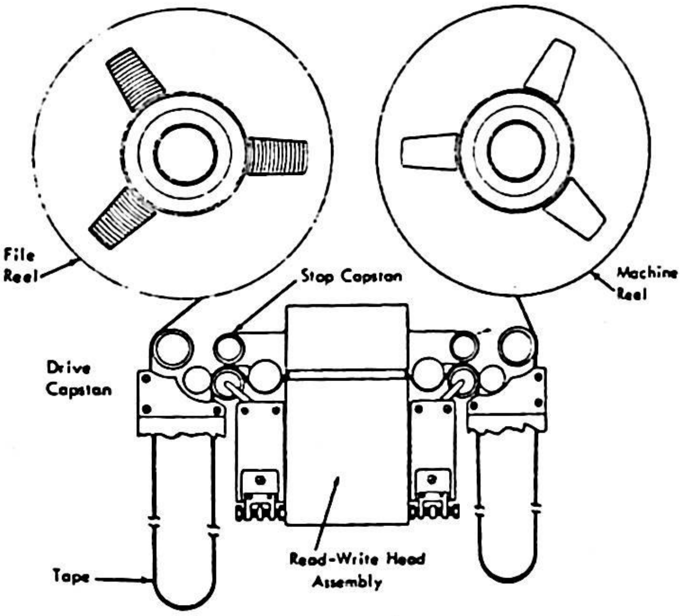


ד. **הפקה של נתונים סופיים:** הסיבות שהביאו להזנה של נתונים לתהליכים באמצעו. סרטים הם אשר הינחו בהפקתם אל סרט מגנטי וממנו, בתהליך נפרד, למדפסת. ההפקה באמצעות מדפסת הקשורה ישירות לתוכנית מקובלת במחשבים קטנים ובינוניים בלבד, ולא במחשבים הפועלים ב-Multiprogramming.

הנתונים מוקלטים בסרט מגנטי בשיטה מגנטית בקוים זעירים לרוחב הסרט. לכל



כונני סרטים מגנטיים



תאור טכני של כונן סרט מגנטי (י.ב.מ.).

נקודת מיגנוט כזו קוראים "סבית". ההקלטה הזו נשמרת לזמן רב וניתנת ל"קריאה" ע"י מתקן הסרט המגנטי פעמים רבות. בזמן ההקלטה הבאה של נתונים על אותו סרט נרשמים הנתונים החדשים על הקודמים ללא צורך ב"מחיקה" של הנתונים הקודמים. התוים נרשמים לרוחב הסרט (ע"י צרוף של סביות) בצפיפות של 800 או 1,600 תוים לאינטש בהתאם למבנה ולדרישות המתקן.

הנתונים נרשמים כתוים אלפנומריים או חלק ממלת נתונים בינרית. לגבי מתקן הסרט אין כל הבדל בכך מכיון שהוא קולט ומזין נתונים במספר סביות קבוע, בהתאם למספר הערוצים בסרט. בסרטים המגנטיים הנפוצים קיימים 7 או 9 ערוצי רישום של נתונים. 6 או 8 מיועדים לרישום הסביות של תו אחד והערוץ הנוסף משמש לבקורת מתום (PARITY). בזמן רישום הנתונים בסרט משלים המתקן את מספר הסביות בכל תו למספר זוגי או אי-זוגי כפי שנדרש. בזמן הקריאה נבדקת מהימנות הקריאה באופן אוטומטי ע"י בקורת מספר הסביות בשיטה דומה.

רישום התוים על הסרט איננו רציף לגמרי (כפי שנעשה הדבר בסרט הנייר למשל), אלא נעשה בקבוצות כאשר ביניהן אזור ריק, לא מוקלט. הסיבה לכך נעוצה במהירות התנועה המקובלת של הסרט הנמצאת בתחום 75 עד 200 אינטש לשניה. מתקן הסרט דורש מרחק מסויים לתאוצה לפני התחלת העברת הנתונים ומרחק עצירה, כאשר הוא נדרש להפסיק את העברתם. אורך המירווח בין גושי נתונים קבוע ונע בין 0.5 אינטש לאינטש אחד בהתאם למתקן. בסוף כל גוש נתונים יש בדיקת מתום לאורך המאפשרת בקורת נוספת על שלימות הקריאה. סביות המתום נרשמות באופן אוטומטי בזמן כתיבת הנתונים ונבדקות אוטומטית בזמן קריאתם.

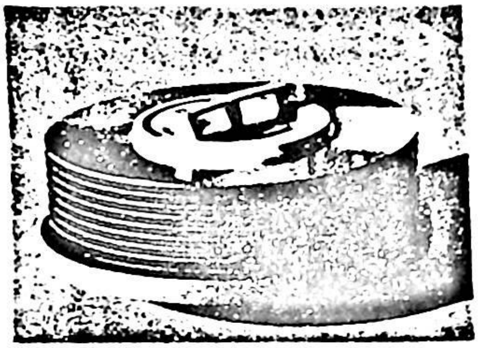
הסרטים המגנטיים מגוללים בגלילים ברוחב ½ אינטש ובאורך של 2,400 רגל. קבולת סרט מגנטי עם גושי נתונים של 2000 תוים בצפיפות של 556 תוים לאינטש היא 13 מיליון תוים, בצפיפות של 800 הקבולת היא 18 מיליון תוים ובצפיפות של 1600 תוים לאינטש הקבולת מגיעה ל-31 מיליון תוים.

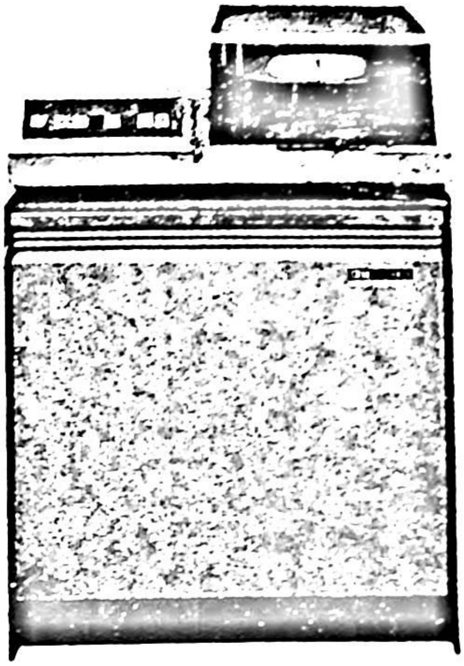
מהירות העברת נתונים בסרטים מגנטיים נעה בתחום שבין 15,000 תוים עד 340,000 תוים בשניה, ואף מגיעה ל-1,250,000 תוים בשניה.

אחסון לגישה ישירה

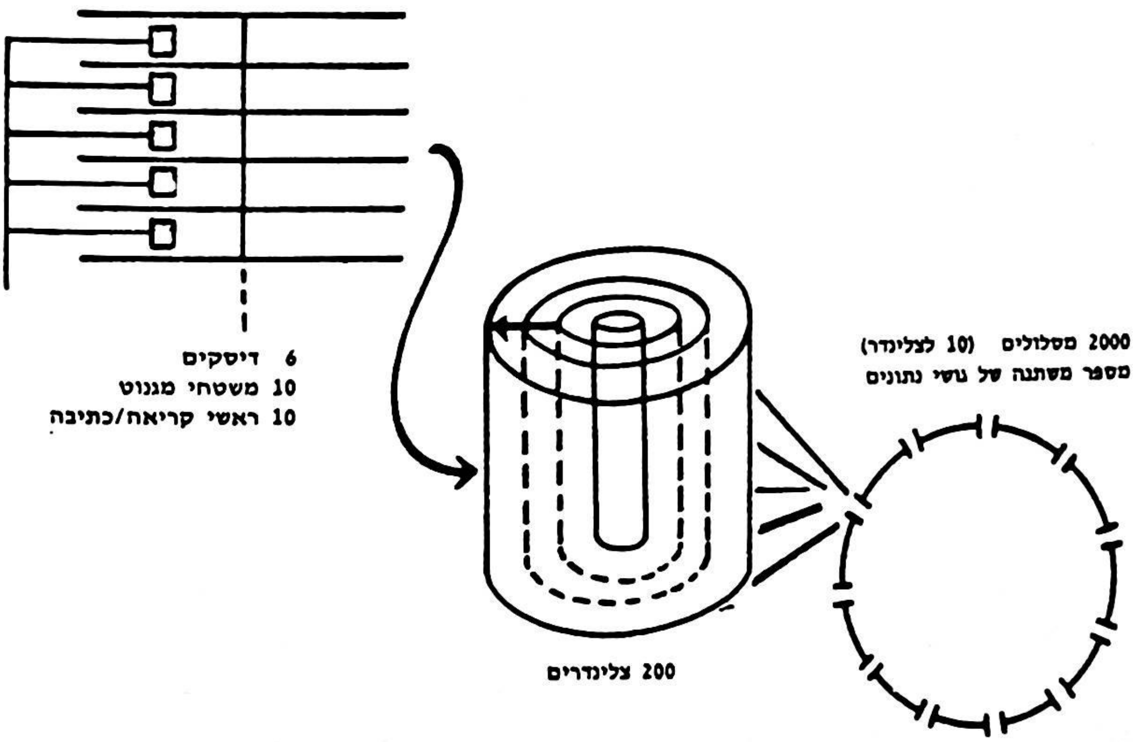
הסרט המגנטי מקובל על כל החברות המייצרות מחשבים כאמצעי אחסון של נתונים. בנוסף לו פותחו מתקנים נוספים המבוססים גם הם על שיטת הרישום המגנטית, אולם שיטת פעולתם וצורת הרישום בהם שונה. המטרה העיקרית של פתוח המתקנים האלה הינה גישה ישירה אל כל נתון הרשום במתקן האחסון.

את האפשרות של גישה אל כל נתון אפשר להדגים על-ידי התקליט בפטיפון לעומת





מערכת תקליטים מגנטיים



תאור תכולת מתקן תקליטים

סרט מגנטי ברשמקול. בתקליטים גדולים מקובל להקליט מספר רב של שירים בזה אחר זה. לכל שיר מוקצב מספר מסלולים קבוע ואנו יכולים להשמיע את השיר השלישי למשל על-ידי כוון הכפתור למספר שלוש. הזרוע של הפטיפון תנוע אל המקום המדוייק בו מתחיל השיר השלישי, כפי שדרשנו. לא כן המצב בסרט של רשמקול, עלינו להשמיע את השיר הראשון והשני כדי להגיע אל השיר השלישי. כדי להתגבר על כך הותקן מד-סרט ואפשרות של דילוג ללא השמעת השירים אולם אלו הן וריאציות על סריקה רצופה של השירים עד למקום המבוקש.

דוגמה נוספת, אם כי לא זהה לגמרי, יכולה להיות בכרטסת יד בעלת חוצצים לכל קבוצת כרטיסים. כאשר מחפשים כרטיס מסוים אפשר לחשב את מקומו בכרטסת ולפנות ישירות אל הקבוצה בה הוא נמצא ורק אז לסרוק באופן רצוף כל כרטיס בקבוצה.

איך ישימה שיטת האחסון במתקני האחסון של המחשב?

אחסון נתונים רציף (Sequential Storage Devices)

שיטת האחסון הרציפה קיימת בכל מתקני האחסון המגנטיים (וגם בכרטסות של כרטיסים מנוקבים וסרטי נייר מנוקבים). הנתונים נרשמים במתקן האחסון לפי סדר הפקתם בתהליך המחשב ועל כן גם אפשרות הקריאה הינה ברציפות.

לדוגמה: כרטסת תושבים מסודרת לפי מספרי תעודת הזהוי. בכדי להגיע לנתונים השייכים לתושב כל שהוא יש להתקדם ולפסוח על הנתונים של כל התושבים שמספר הזהוי שלהם קטן ממספר הזהוי המבוקש. פעולה זו כרוכה בקריאת הנתונים ממתקן האחסון של המחשב, ופעולת השואה במחשב של מספרי הזהוי המתקבלים אל מספר הזהוי הנתון. כתוצאה מפעולת ההשואה נמשיך בקריאה אם מספרי הזהוי קטנים מהמספר המבוקש, נפסיק את הפעולה אם הם גדולים ממנו (הוא אינו קיים בכרטסת) או נבצע את התהליך כאשר מספר שנקרא שוה למספר המבוקש. אפשר לראות מדוגמה זו ששיטת העיבוד הזו מחייבת מיון הנתונים המוזנים למטרות עדכון של נתונים במתקן אחסון רציף, לפי סדר הנתונים המאוחסנים.

אחסון נתונים בגישה ישירה (Direct Access Storage Devices)

שיטת האחסון הזו קיימת במתקנים מסוג הדיסק המגנטי או התוף המגנטי, בהם הנתונים מסודרים במקומות קבועים והגישה אליהם נעשית באופן ישיר לפי "כתובתם". יש גמישות רבה בשיטות הגישה הישירה אשר אינה תלויה ביכולת המתקן בלבד אלא מבוססת גם על תכניות מתאימות במטרה להקל ולפשט את הטפול בנתונים: קיימת אפשרות לגישה ישירה ממש ע"י הצבת כל נתון במקום קבוע; בשיטה אחרת ניגשים לאזור שבו

נמצא הנתון המבוקש ואז סורקים ברציפות אזור קטן יחסית של המתקן; שיטה נוספת היא לרשום את הנתונים ברציפות ולנהל אינדקס מראי-מקום, ואז לבצע את הפניה לנתון בשני שלבים, קודם חפוש כתבתו באינדקס ואח"כ חפוש הנתונים עצמם בגישה ישירה.

אחסון הנתונים בגישה ישירה יכול להתבצע כאשר אפשר לתת כתובות לאזורים שונים במתקן האחסון. בנוסף לכך צריך להיות אמצעי המאפשר לראש הקריאה/כתיבה להגיעץ א כל נתון, באופן ישיר ובמהירות, מבלי שיצטרך לסרוק את הקובץ לשם כך.

יתרונותיה של שיטת האחסון בגישה ישירה רבים: היא מאפשרת עיבוד נתונים בודדים מתוך אוסף נתונים רב בזמן קצר באופן יחסי לזמן שהיה נדרש לעיבודם ב- שיטת האחסון הרציף.

קריאה מגנטית וקריאה אופטית

השאיפה לקלוט במחשב מסמכי מקור, בכדי לחסוך בהוצאות הסבתם לכרטיסים או סרטים מנוקבים ובכדי להגיע למהימנות גבוהה יותר, מצאה את פתרונה בהפעלתם של קרוא הסימנים האופטי והקורא המגנטי.

שיטת הקריאה המגנטית מבוססת על רישום התוים על נייר בדיו מיוחדת ובתב- נית מוגדרת שאותה יכולה המכונה לזהות ע"י גישוש. לצורך רישום הנתונים בדיו מגנטית משתמשים במדפסות מיוחדות או במתקני הטבעה המשמשים כאן כמכונות נקוב כרטיסים מנוקבים. גודל המסמכים המשמשים לקריאה מגנטית זהה בדרך כלל לגודל כרטיס מנוקב.

חשיבותם של המסמכים לקריאה מגנטית בכך שניתן להטביעם מראש בבית דפוס ולהוסיף בעת הצורך רק נתון אחד או שנים כדי להשלים את כל הפרטים לפני העבר- תם למחשב לעיבוד. השמוש בדיו מגנטית מקובל בבנקים עבור פנקסי השיקים: כל פרטי השיק מוטבעים מראש, כולל מספר החשבון. כאשר השיק נפרע מוסיפים רק את הסכום ע"י הטבעה בבנק עצמו ושולחים אותו למסלקה לעיבוד נוסף (מיון, קלט למח- שב).

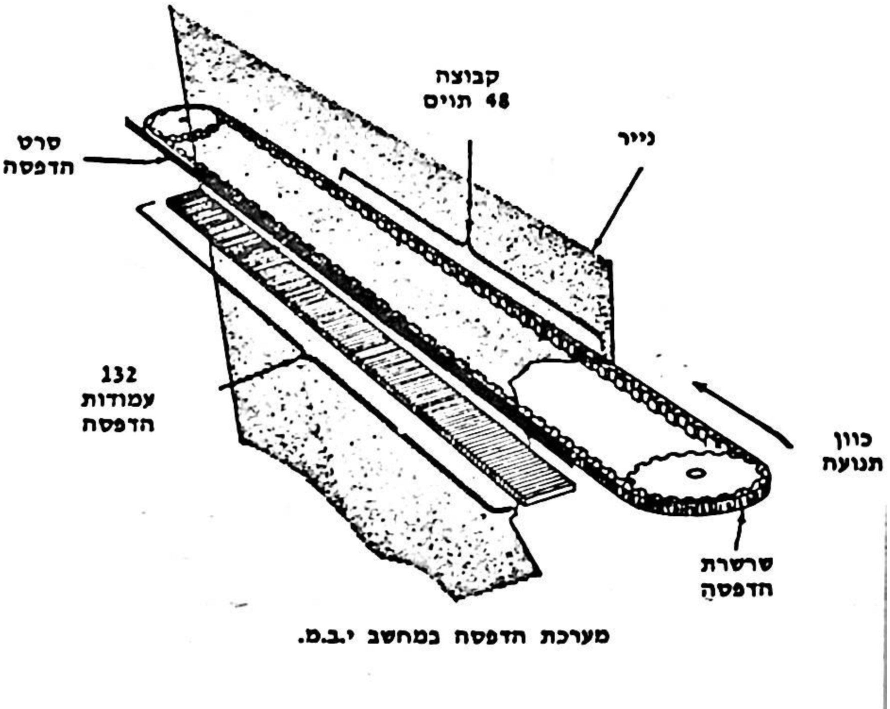
שיטת הקריאה האופטית שונה משיטת הקריאה המגנטית. היא באה לפתור את בעית הקלט למחשב, במיוחד במסמכים שהודפסו ע"י המחשב כהודעות ללקוחות לתשלום, וחזרו אל המחשב כאשורי תשלום לאחר שהפעולה בוצעה (תשלום חשמל, אגרות וכד'). מתקני הקריאה האופטיים יכולים לקרוא רישום עיפרון והדפסת מכונה ומוגבלים לספ- רות ומספר תוים נוספים בלבד.

המדפסת

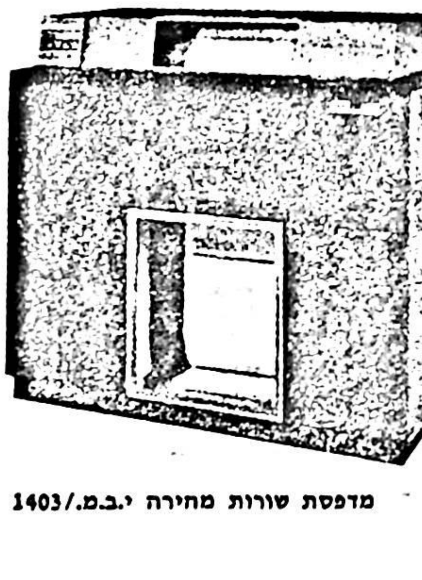
המדפסת היא אמצעי הפלט החשוב ביותר מכיון שהיא מעבירה את המוצר המוגמר של תהליך העיבוד לצורה שניתנת לקריאה, לעיון ולהערכה ע"י בני-אדם.

מבחינה טכנית מורכבת מערכת ההדפסה המקובלת ממנגנון עם מסילה שבה נעות חוליות שבכל אחת מהן חרוטים שלשה תוים מתוך האותיות, הספרות והסימנים של השפה שבה מדפיסים (עברית או אנגלית). לעומת המסילה עם החוליות נמצאים מקשים, מקש אחד לכל עמודת הדפסה אפשרית. כלומר, מספר התוים בשורת ההדפסה שוה למספר המקשים. (בדרך כלל כ-130). הנייר שעליו מדפיסים וסרט הפחם נמצאים בין המקשים למסילת החוליות ("שרשרת"). בזמן שמתבצעת ההדפסה לפי הוראת תוכנית המחשב, מכה המקש המתאים אל עבר החוליה הנמצאת מולו באותו זמן וכך נוצרת תמונת האות הרצויה על הנייר. החוליות נושאות תוים שונים והן נעות כל הזמן במהירות במסילה, ועל כן נדרשת סינכרוניזציה בין האות הנמצאת בעמודת הדפסה מסוימת לבין מועד ההקשה של המקש באותה עמודה. כאשר כל המקשים בעמודות ההדפסה השונות השלימו את פעולתם - הסתימה הדפסת שורה והנייר מקודם לשם הדפסת השורה הבאה. נייר ההדפסה הוא רציף ומאפשר הדפסה של כ-1000 דפים ויותר ללא עצירה. קצב ההדפסה - כ-1000 עד 2000 שורות בדקה.

המדפסת מבצעת פעולה אוטומטית של הדפסת שורות הנשלחות אליה מהמחשב, ולכן התוכנית צריכה להכין את מבנה השורה ותוכנה בצורה אשר תאפשר קריאה נוחה. כלומר, הנתונים צריכים להיות ערוכים עם כותרות בסדר הגיוני, בפיסוק מתאים, רווחים בין שדות, עם סיכומים וסיכומי ביניים ועוד.



מערכת הדפסה במחשב י.ב.מ.



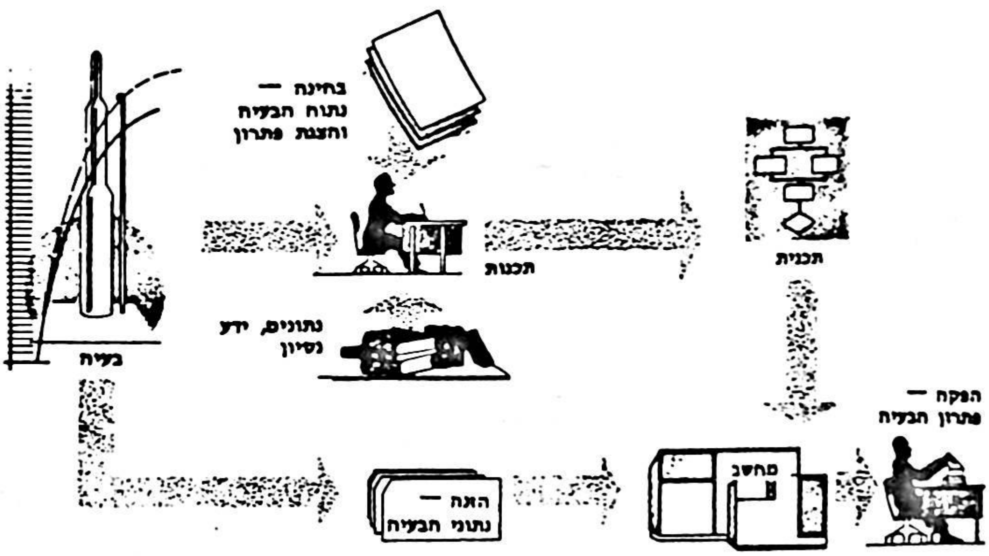
מדפסת שורות מהירה י.ב.מ./1403

פרק 2: תהליכי תכנון ותכנות

התהליכים של תכנון ופתוח מערכת תוכנה מכוונים לבניית אחד המרכיבים העיקריים של מערכת עיבוד הנתונים - התוכנה (Software). תהליכים אלו כוללים מספר שלבים, המתחילים בשלב הצגת הצורך למיכון תהליכי עיבוד נתונים ומסתיים בשלב ההפעלה של המערכת לפי דרישותיו של הצרכן המזמין אותה.

שלבים בהפעלת מערכת עיבוד נתונים

השלב הראשון בתהליך הוא שלב סקר הישימות (feasibility study) שבו מוג-דרת הבעיה שלשמה מתכוונים להפעיל מערכת עיבוד נתונים. בשלב זה נבחנת גם האפ-שרות להשיג את המטרה בשיטות מיכון במחשב וכדאיות הפעלת המיכון. השלב הבא הוא הבחינה (system analysis) שבו עורכים בדיקה מפורטת של הנושא שמתכוונים להפעיל במחשב (ניהול מלאי, תהליכי גבית תשלומים, סקר דעת קהל וכדי') ומציגים פתרון. זהו תכנון מפורט של השיטה לבצוע העבודה במיכון הכולל הגדרת הקלט, הפלט, פעו-לות העיבוד והקבצים. לאחר השלמת התכנון נעשה התכנות (programming), שהוא למעשה מתן ההוראות למחשב לבצע את פעולות העיבוד הנדרשות. את התוכניות (programs) הנכתבות מנפים משגיאות (debugging) ובודקים עם נתוני נסוי (test data) כאשר הפעלת התוכניות עם נתוני נסוי נותנת את התוצאות המבוקשות "מרי-צים" את התוכנית עם נתוני קלט אמיתיים. שלב זה, אם עבר בהצלחה הינו שלב הפעלת המערכת שבו עוברים הנתונים מתוכנית העיבוד הראשונה הקולטת את הנתונים ועד לאחרונה - המפיקה את הדוחות הרצויים.



שלבי פתרון בעיה בעזרת מחשב

בחינה

ההצעה לשיטת עיבוד הנתונים מוצגת במסמך "תיק הבחינה" או "אפיון המער-כת". לאחר אישורו פונים לתכנון מפורט של מערכת עיבוד הנתונים, תכנות והפעלה. תיק הבחינה מחולק לשני חלקים עיקריים:

- תיק התהליך;
- תיקי המהלכים.

א. **תיק התהליך** מתאר את הנושא בכללו ומפרט את השלבים השונים (המהלכים) שיש לבצע כדי להשלים את כל העבודה הדרושה למלוי המשימה. תיק התהליך משמש ל-תאום בין המהלכים השונים בתהליך ומפרט את מערך זרימת הנתונים בשלבי הקלט, העיבוד והפלט. מקובל להציג את התהליך באמצעות תרשים זרימה (Flow Chart). התרשים הוא אמצעי נוח לתאור הפתרון הלוגי לבעיה וממחיש את הקשרים שבין המשימות והשלבים השונים בבצוע התהליך. שיטת ההצגה ע"י תרשימים מהוה דו-קומנטציה סטנדרטית, מסודרת ונוחה להבנה ולשמוש.

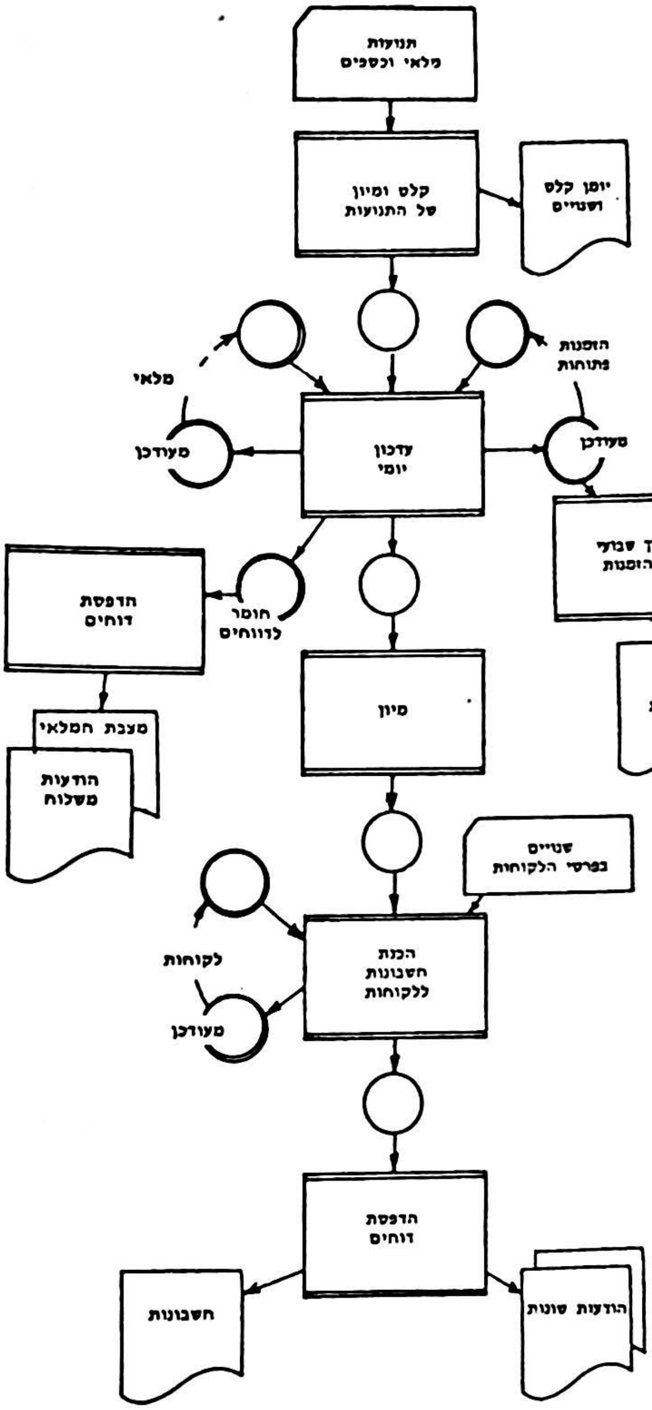
ב. **תיקי המהלכים** הינם מסמך תעודתי מדוייק: הם מהוים פרוט לתיק התהליך ומיו-עדים לשמש בסיס לתכנות ומקור נתונים יחיד אודות כל מהלך. תיק המהלך מפרט את כל הנחות היסוד עליהן מושתת המהלך ומספק את כל הנתונים וההסברים הדרו-שים, לעתים גם בעזרת דיאגרמות, תרשימים ונוסחאות. בחלקו השני הוא מפרט בעזרת תרשימים טפסים את שלבי התכנית, צורה ומבנה של נתוני קלט/פלט ומציין את יחידות המחשב המשתתפות בבצוע. ההצגה בתיק המהלך נעשית בעיקר בתרשים המלבני (Block Diagram) הניתן כאן בצורה פונקציונלית של תאור הקשר הלוגי בין הפעולות השונות לפתרון הבעיה.

תיקי בחינה - סכום השיטה המוצעת

השיטה המוצעת ע"י הבוחנים מוצאת את בטויה בתיקי הבחינה. התאור הכללי של השיטה ניתן בתאור ה"תהליך" של העיבוד. התהליך מפורט ל"מהלכים" - שלבים בעיבוד הנתונים - שיש בהם פעולת קלט אחת ופעולת פלט אחת.

בתאור מהלך יש לציין את הפרטים הבאים:

דוגמת תרשים תהליך
לעדכון מלאי מחסנים והכנת חשבונות לקוחות



<u>קבצים:</u>

א. <u>קבצי אב</u> בסרטים מגנטיים: מלאי, הזמנות פתוחות, לקוהות.

ב. <u>קובץ תנועה</u> בכרטיסים מנוקבים.

ג. <u>קבצי פלט</u> במדפסת.

ד. <u>קבצי בינים</u> בסרטים מגנטיים..

א. מטרה - מטרת תהליך העיבוד ומקומו בתהליך עיבוד הנתונים.

ב. קלט - תאור הנתונים המתקבלים, שיטת הבדיקה וצורת עריכתם להמשך העיבוד.

ג. פלט - צורת עריכת הפלט (דו"חים) המיועד להדפסה ולמסירה לבני אדם, ומבנה הפלט המיועד למהלכי עיבוד נוספים.

ד. עיבוד - הפעילויות הלוגיות המתבצעות בתהליך עיבוד הנתונים במהלך זה.

ה. קבצים - תכולת "כרטסות הנתונים" של המערכת. תאור שיטת העדכון.

את תהליך העיבוד מקובל להציג בעזרת תרשימי זרימה. את הקבצים מתארים לפי האלמנטים המרכיבים אותם.

תרשימי זרימה

תרשים זרימה של שיטה הינו הצגה גרפית של שיטת עיבוד ההופכת נתונים ממסמכי מקור ונתונים בסיסיים למסמכים סופיים.
התרשימים השונים באים לשרת מטרות אחדות:

1. <u>תרשים התהליך</u> המבוצע ע"י הבוחן מתאר את השלבים בבצוע המשימה, מפרט את עיקר נתוני הקלט והפלט ואת הקבצים המשתתפים בכל מהלך ומהלך.
2. <u>תרשים המהלך</u> המבוצע ע"י המתכנת מפרט את העיבוד בכל מהלך. המתכנת מסתמך בעבודתו על תרשים מהלך כללי-פונקציונאלי, שעיקרו - הפתרון הלוגי של הבעיה.

חשיבותו של התרשים בכך שהוא:

1. עזרה לפתוח התכנית - מסייע לנתוח הבעיות הלוגיות על ידי הצגת השלבים הלוגיים לפתרון.
2. מסמך של התכנית - מוסר את עיקרי הנושאים בצורה ברורה וחד-משמעית.
3. הדרכה ועזר בקידוד (כתיבת ההוראות) - כתיבת התכנית נעשית על פי התרשים.

יש להדגיש את חשיבותם של התרשימים השונים בהיותם דרך הצגה אחידה, מוסכמת ומקובלת על כל העוסקים בעיבוד נתונים במחשבים.

תאור קבצים

<u>שדה</u> (Field) - נתון מוגדר כלשהוא.

<u>רשומה</u> (Record) - נתון אחד או יותר המהוים שלימות אחת: "רשומת עובד" מכילה את שם העובד, כתובתו, מספרו האישי, מצבו המשפחתי וכד'.
רשומת פריט במחסן מכילה את מספרו הקטלוגי, מחיר הקניה שלו, מחיר המכירה, כמות במלאי, נקודת הזמנה וכד'.

קובץ (File) - אוסף הרשומות המתיחסות לנושא אחד. למשל: קובץ פריטים במחסן, קובץ מרשם התושבים המכיל את רשומות כל התושבים בארץ, קובץ סטודנטים וכד'.

תכנות והידור

מה זה תכנות?

נהוג לעתים לקרוא למחשב "מוח אלקטרוני". אולם זהו מוח טפש, חסר אינ-טליגנציה, כושר מחשבה ודמיון, וחסר כושר יצירה. מערכת המחשב יכולה לבצע הו-ראות במהירות עצומה, היא יכולה לבצע בקורות ולציין שגיאות בפעולתה התקינה. אולם המחשב איננו יכול לעלות על מוח האדם, אשר בשלב זה של התפתחות המחשבים עדיין מכוון אותו בפעולתו, ומבצע עבורו את החשיבה המוקדמת. נוכל להגדיר את המונח "מחשב" בצורה הבאה: "התקן המסוגל לקלוט נתונים, לעבדם לפי תכנית מוכ-תבת ולספק את תוצאות העיבוד"; אולם לא נשתמש במושג "מוח" או "חשיבה".

פיתוח התכונות הפיסיות של המחשב אינו מאפשר עדיין את נצולו. המחשב ניתן להפעלה ע"י הוראות הנמצאות בזכרון הפנימי ומוזנות אוטומטית לפי סדר אל יחידת הבקרה המפענחת אותן ופועלת לפיהן. מערכת ההוראות לבצוע סדרת פעולות במחשב נקראת "תכנית", והכנתן - "תכנות" (Programming).

שפת מחשב ושפת תכנות

שפת המחשב היא הצורה בה נמצאות ההוראות בזכרון המחשב. אפשר לכתוב כל תכנית בשפת המחשב ע"י שמוש בקודים שלו עבור הפקודות ושמוש בכתובות זכרון מו-חלטות. שיטה זו אינה טובה ויעילה לכתיבת תכניות גדולות ומשתמשים בה לעתים לתכניות בעלות מספר קטן של הוראות בלבד. יש לזכור שבמחשבים הראשונים שנבנו היתה זו הדרך היחידה להכנת תכניות והזנתן לזכרון.

כתיבת תוכניות בשפת המחשב הינה פעולה קשה וממושכת הצופנת אפשרויות רבות לשגיאות. ככל שהמחשב משוכלל יותר וככל שדרישות הבצוע מורכבות ומסובכות יותר, נעשית כתיבת התוכנית ממושכת יותר ויקרה יותר, עד כדי כך שהיא מהוה חלק ניכר מתוך ההוצאה הכללית להחזקת מחשב.

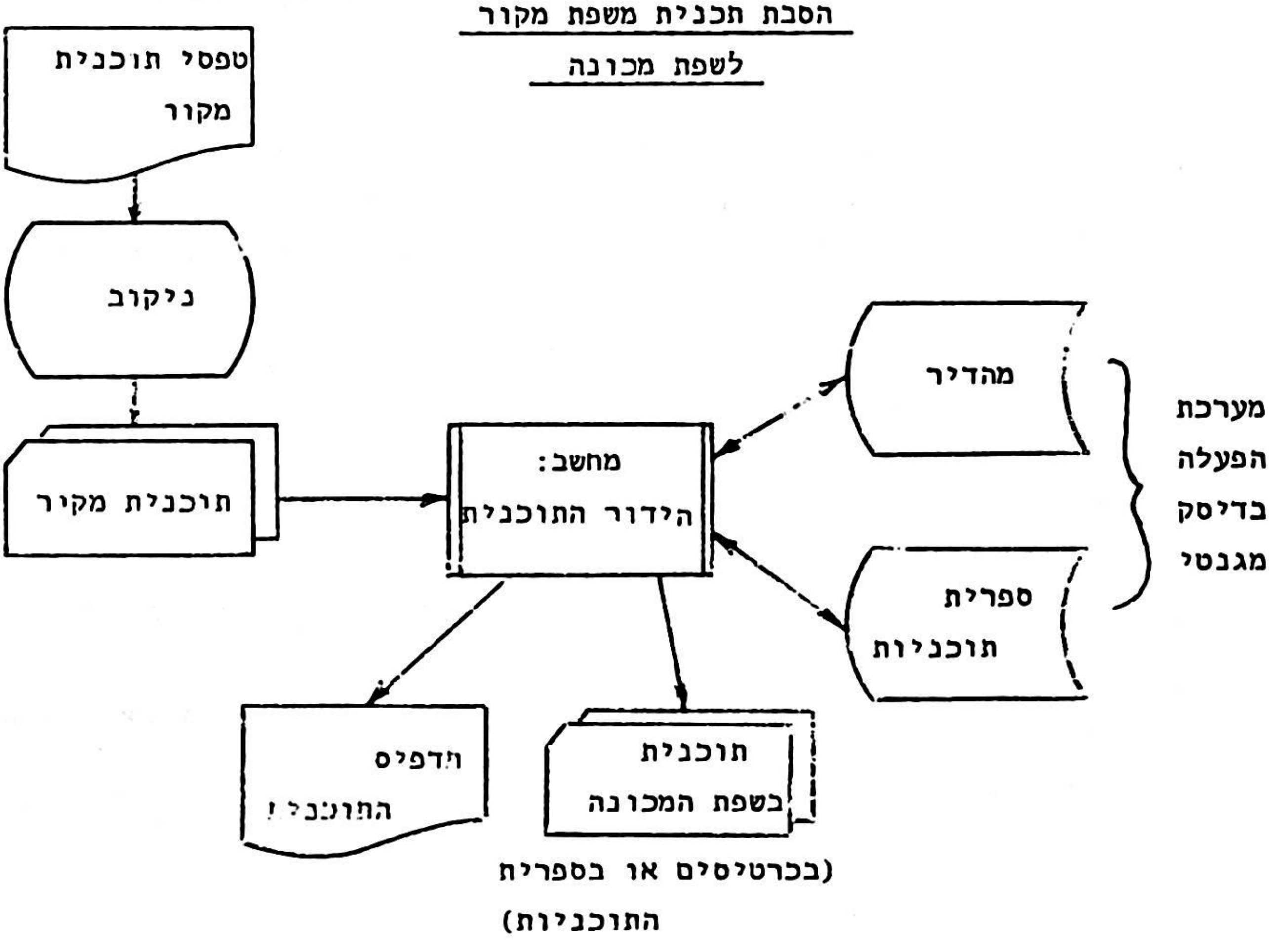
כדי להתגבר על בעיה זו החלו לפתח שפות לשם כתיבת תכניות (Source Program) אשר תהיינה נוחות לשמוש ועם זאת אפשר יהיה להפכן לשפת-מכונה (Object Program) המפעילה את המחשב. הפיכת תכנית בשפת מקור לתכנית בשפת מכונה נעשית בעזרת תכ-נית הידור המתאימה לשפת התכנות שבה נכתבה תכנית המקור. הקלות היחסית של כתי-

בת תכניות באמצעות שפות אלי יצרה מצב בו אין מעלים על הדעת כתיבה בשפת המכ.-נה ורואים שפות אלו כהשלמה הכרחית למכונה. התכונות של המחשב אינן נבחנות אי-פוא רק לפי התכונות הפיסיות של המכונה (Hardware) אלא במדה רבה לפי התכונות של השפות - התוכנה (Software).

קיימות רמות שונות של שפות תכנות (מלשון "תוכנית") בהתאם למידת קירבתן לשפת המכונה או לשפת אנוש (שפות עיליות). בשפות תכנות אלי מקנה התוכניתן ל-הוראות כנויים או ערכים סמליים. תוכנית עיבוד הנקראת <u>מהדיר</u> (Compiler) קוראת את התוכנית שכתב התוכניתן כנתונים והופכת אותה לתוכנית בשפת מחשב כנדרש לשם בצוע.

פעולת ההידור נעשית בכמה שלבים. לצורך הדגמה נבחין בשני השלבים העיקריים:
בשלב הראשון מתרגמת התכנית את ההוראות ומקצה את שטחי הזכרון לפי הסדר שהוג-דרו, החל מכתובת הזכרון שנקבעה ע"י המתכנת. באות: זמן היא מכינה טבלת התיחסות של התויות לכתובות הזכרון. בשלב שני היא מחליפה בגוף כל הוראה את התיות ה-מסמלות כתובת, בכתובות הזכרון אשר נלקחו מהטבלה אשר הוכנה בשלב הראשון.

<u>הסבת תכנית משפת מקור</u>
<u>לשפת מכונה</u>



תהליך הכנת תוכנית

תהליך הכנת תכנית למחשב כולל שלשה שלבים: הכנת תרשים לוגי של המהלך (Block Diagram), כתיבת הוראות התכנית (Coding), ונפוי התכניות משגיאות ע"י נסוי במחשב (Debugging).

הכנת התכנית נעשית בשלבים אחדים. בתכנית פשוטה קטן חלקם של השלבים המטפלים בפתרון הלוגי של הבעיה, לעומת חלקם בתכנית מורכבת, שבה הפתרון "ה-תכנותי" הוא חלק נכבד מהמאמץ המושקע בכתיבת התכנית.

א. עריכת תרשים מלבני ובדיקתו

עריכת תרשים מלבני מפורט לפי תרשים המהלך הכללי (פונקציונלי) שבתיק המהלך. בתרשים זה יש לתאר את פעולות המהלך בדרג של הוראות למכונה ואת הש-מוש בשגרות שבאות מספרית התכניות הכללית אושנכהבו על ידי התכניתן לצורך ה-תכנית. כמו כן יש לפרט את נוסחאות החשוב לשלביהן. תרשים מלבני שמיועד ל-תכנית שתכתב בשפה יוצרת אינו צריך להיות באותה מידת פירוט כתרשים לכתיבת תכנית בשפה סימלית.

כאשר מסתימת הכנת התרשים יש לערוך "בדיקות שולחן" (Desk Check) כדי לודא מספר דברים:

1. התרשים המלבני תואם את תרשים המהלך הכללי מבחינת השיטה;
2. תאור נכון של הנוסחאות ושמוש בקודים ובסמולים כפי שנדרש;
3. בצוע כל הפעולות האחרות שנדרשו בתרשים המהלך הכללי;
4. הבדיקות הלוגיות אשר בוצעו בתרשים - נכונות (פעולות ההשואה והמסקנות הנובעות מהן).

הבדיקה נעשית ע"י בחירת כווני פעולה שונים בתרשים, ומעקב צעד צעד ב-עזרת מספר דוגמאות של נתונים. יש להדגיש שבשלב זה על התכניתן להחליט החל-טות שונות הקשורות בארגון האינפורמציה ובבצוע בדיקות שונות ותגובום עליהן. בחלק מהמקרים הקשורים בשיטה הוא פונה לעזרת הבוחן. תוך כדי הכנת התרשים ה-מפורט חייב התכניתן לנהל רשימות לגבי הדברים האמורים לעיל, כדי שניתן יהיה להסתייע בהם בשעת הפעלת התכנית

ב. כתיבת התכנית והידור

התכנית נכתבת בשפת התכנות של המחשב שבו היא תופעל. שלב הכתיבה

של התוכנית, הקידוד, היא רישום ההוראות לפי התרשים המלבני את ההוראות רוש-מים בטופס קידוד המתאים ל"שפה" שבה נכתבת התוכנית. כל שורה בטופס מיועדת לניקוב בכרטיס אחד, ועל כן מספר העמודות בשורה הוא 80 - עמודה לכל טור בכרטיס. לאחר שכותבים את התוכנית מנקבים אותה בכרטיסים ומזינים איתם למחשב שבו מופעל מהדיר (compiler) המתאים לשפה שבה נכתבה התוכנית, למשל FORTRAN, COBOL או PL/I. המהדיר מתיחס אל תוכנית-המקור כאל נתונים; תהליך העיבוד שלו הוא הפקת תוכנית בשפת מכונה.

המהדיר בודק את תוכנו של כל כרטיס; אם אינו "מבין" הוראה כלשהי (בדרך כלל מפני שההוראה לא נכתבה בהתאם לכללי שפת התכנות) הוא מודיע בתדפיס התוכ-נית על קיום שגיאה. המתכנת מתקן את השגיאות ומגיש את התוכנית להידור נוסף. כאשר אין יותר שגיאות מפיק המהדיר את התוכנית בשפת מכונה, שאותה אפשר לקבל בכרטיסים מנוקבים או בדיסק מגנטי בקובץ (ספרית) התוכניות. התוכנית בשפת המכונה מיועדת להזנה למחשב לבצוע.

ג. הרצות נפוי התוכנית - Debugging

נפוי התוכנית פרושו שיש צורך לבצע בדיקה של התוכנית ול"נקות" ממנה שגיאות אפשריות. כלומר, למרות שהתוכנית תורגמה ע"י המהדיר ללא שגיאות יתכן שהיא לא תפיק את התוצאות המבוקשות. המחשב מבצע כל הוראה כפי שהיא נכתבת, ועל המתכנת לדאוג לכך שהוראות התוכנית שלו תהיינה משמעותיות ותשתלבנה בתהליך עיבוד נכון. אם המתכנת מתכוון להדפיס שם של אדם ו"בטעות" כתב הוראה להדפסת גילו או כתובתו תתקבל תוצאה שאינה נכונה. כדומה אם רצה לכפול שני מספרים ובטעות חיברם אותם. המחשב <u>אינו</u> יכול לגלות שגיאה מסוג זה. לפעמים כתובה הו-ראה שאינה ניתנת לבצוע כמו למשל חילוק באפס. במקרה הראשון - תודפס תוצאה שאינה רצויה ובפעם השניה - יודיע המחשב על "שגיאת בצוע" (program check) ויפסיק את בצוע התוכנית. המהדיר לא יכול לגלות מצב של חילוק באפס מכיון שהנתונים מתקבלים רק בשלב הבצוע. בתהליך הנפוי מבצעים הלכה למעשה את כל הפעולות והבדיקות הנדרשות בתוכנית, גם אלו שהן יוצאות דופן ועתידות להתבצע בחומר אמיתי רק לעתים נדירות. יש לטפל בכל סוגי הקלט שהמהלך עתיד לקבל, ולבצע את כל סוגי הפלט על יחידות האחסון המגנטי ובהדפסה. יש להזין קלט עם שגיאות מכוונות לבדיקת עצירת התוכנית וההתרעה על שגיאות.

כמות החומר היא קטנה באופן יחסי (עשרות או מאות כרטיסים או רשומות),

אולם מייצגת, על כן הכנתו לנסוי היא פעולה אחראית וחשובה. את הפלט של הרצות הנסוי צריך לבדוק בקפדנות רבה, כי לא כל חומר מודפס ומנוקב במבנה נכון, הוא גם נכון בתוכנו. סדרת הרצות הנסוי משמשת לנפוי התוכנית משגיאות, או בשפה המ-קובלת - Debugging והנתונים המשמשים לנסוי - Test-Data.

ד. נסוי המהלך כחלק מתהליך שלם

תהליך עיבוד נתונים לפתרון בעיה מסוימת מורכב בדרך כלל ממספר מהלכים/ תוכניות. לאחר בדיקה ונפוי של כל תוכנית בנפרד יש לבדוק רצף שלם של תוכניות, החל מהתוכנית הראשונה המקבלת את הנתונים ועד התוכנית האחרונה המפיקה את הפלט הסופי.

בתהליך עיבוד קובץ יתרות חשבונות בבנק יכולים להיות המהלכים הבאים: תוכ-נית קלט של התנועות (משיכות והפקדות), תוכנית מיון, תוכנית עדכון קובץ היתרות, תוכנית הדפסת דפי חשבון. פלט של המהלך הראשון (קלט) מועבר למהלך הבא לעיבוד וכן הלאה, עד להפקת הדווח הסופי.

בתחילה נבדק התהליך עם נתוני נסוי אשר הוכנו ע"י התוכניתן או הבוחן. לאחר שנסוי זה עובר בהצלחה מבצעים "הרצה" של התהליך עם נתונים אמיתיים ולאחריה - הפעלת מערכת עיבוד הנתונים לבציע העבודה הדרושה.

שפות תכנות

שפות תכנות סמליות

השפות המקובלות ביותר הן אלו הנקראות ASSEMBLY LANGUAGE או PROCESSORS שיטת הכתיבה בשפה כזו נקראת "כתיבה בקוד מנמוני" (Mnemonic), קוד נוח לזכרון בשפות אלו משתמשים בסמלים וקצורים בעלי משמעות ליצוג הפקודות ובשמות סמליים משתמשים לציון הכתובות. בזמן כתיבת התוכנית קובע התוכניתן שמות סמליים (תויות - Labels) לקטעי תוכניות, לשדות ולנתונים, מונים וטבלאות אשר בזכרון המחשב. הפניה לנתון או לקטע התוכנית נעשית לפי התוית שנקבעה לו. התוכנית המתרגמת מקנה לתוית ערך מספרי, בהתאם לכתובת ההוראות והנתונים בזכרון המחשב.

לתויות אפשר לתת מובן הקשור בתפקידם כגון "יתרה", "תנועה", "סימן בקורת", "כתובת" (התוית ניתנת בשפה האנגלית).

קטעים רבים בתוכנית בשפת מכונה מורכבים מהוראות אחדות שבאות למלא משימה אחת כגון כפל שני מספרים, עריכת נתון מספרי להדפסה ועוד. נראה היה שאפשר יהיה להכליל מספר הוראות אלו בהוראה כללית אחת בכדי לחסוך בכתיבה ולהקטין את מספר השגיאות ואז נבנו כלים לצורך זה.

המהדיר נותן לתוכניתן אפשרויות שונות לנוחיות ולחסכון בכתיבה ע"י שימוש בספרית תוכניות קבועה ל"איסוף" תוכניות נוספות ושילובן בתוכנית. בספריה זו מוחזקות תוכניות סטנדרטיות המבצעות פעולות מוגדרות כגון: בדיקת נומריות של נתונים, קריאה של סרט מגנטי ובדיקה של שגיאות קריאה אפשריות, העלאת מספר בחזקה וכיו"ב. תוכניות אלו נקראות שגרות (subroutines).

<u>שפות יוצרות</u>

השפות היוצרות (GENERATORS) נקראות כך משום שבנגוד לשפות התוכנית הסמ-לית אין הן מתרגמות הוראות אלא יוצרות אותן על פי הנחיות או משפטים לוגיים. שפות אלו נקראות גם שפות תכנות עיליות (High level languages).

במרוצת השנים שעברו מאז הופעל המחשב הראשון בחוגות שכוונו ע"י המפעיל, הופעלו שיטות ונוסו דרכים שונות להכנת התוכנית למחשב. המטרה שהנחתה את המתכ-נים בשטח זה היתה להתרחק ככל האפשר מצורת הכתיבה בשפת מכונה, או בשפה סמלית (Assembly), שהיא דמוית-שפת-מכונה. השאיפה היתה להעצר בשלב של נסוח הבעיה ע"י התרשים הפונקציונלי ולהפוך אותו לתוכנית מבלי לרדת לפירוט נוסף. כלומר לכתוב את הפעולה הדרושה ב"שפת בני-אדם" תוך כדי שימוש בשמות סמליים

לערכים ולכתובות בזכרון המחשב. ע"י כתיבה בשפת בני-אדם נראה שאפשר למנוע את רוב הטעויות הטכניות הכרוכות בכתיבת התכניות ולהתרכז בפתרון הבעיה הלו-גית. ואמנם נוחה וגם ברורה צורת כתיבה זו של תכנית בשפת COBOL, למשל:

COMPUTE RESULT = (NUMBER1 + NUMBER2) / NUM3.

כאשר יש לחבר שני מספרים ולחלק את התוצאה במספר שלישי, ואת תוצאת החלוק להעביר לשדה ששמו RESULT.

או .IF A = B ADD 1 TO A AND 2 TO B , ELSE GO TO MULTIPLY-ROUTINE

כאשר יש להשוות שני מספרים ואם תוצאת ההשואה היא שויון יש לחבר לראשון-1 ולשני-2. אם התוצאה אינה שויון, יש לעבור לשגרת כפל מסוים.

השפות היוצרות בקבוצה זו מבוססות על מילון בשפה האנגלית הכולל מאות אחדות של מלים ובעל חוקי דקדוק מיוחדים. קיימות למעשה שתי קבוצות של שפות: לכתיבת תכניות מדעיות הוכנו השפות ALGOL, FORTRAN ולכתיבת תכניות לעיבוד נתונים בשטח המינהלי הוכנה שפת COBOL. שפה המשלבת את התכונות העיקריות של שתי הקבוצות האלה היא PL/1 של חברת י.ב.מ..

כתיבת התכניות בשפה האנגלית הקרובה לשפת המתמטיקה או לנסוח מקצועי של בעית עיבוד נתונים דורשת הכשרה קצרה לעומת ההכשרה הדרושה לכתיבה בשפה סמ-לית החסכון בתקופת ההכשרה ובזמן הכתיבה רב וחשוב. השנויים התכופים בתכ-ניות של עיבוד נתונים מכתיבים אף הם שמוש בשפה משוכללת ונוחה להכנסת שנו-יים.

יתרונות השמוש בשפה יוצרת רבים, ונמנה את החשובים שבהם:

1. **רישומים** - אפשר להסתפק בפחות רישומים הקשורים בקידוד ובהסברים אחרים מכיון שהתכנות עצמו נעשה בשפה ברורה;
2. **אחידות** - התכנית ברורה יותר לאנשים המטפלים בתכניות ומעדכנים אותן.
3. **סטנדרטיזציה** - הכתיבה הקצרה והברורה ע"י שפה מאפשרת שליטה על צורת כתי-בת התכניות ע"י אנשים שונים;
4. **מהירות** - כתיבת תכנית בשיטה זו מהירה פי שלוש בערך מכתיבה בשפה סימלית.
5. **זמן פנוי** - תקון שגיאות בתכנית קל ומהיר יותר מכיון שהשגיאות הן לוגיות בעיקר ולא שגיאות הקשורות בכתיבת התכנית עצמה;
6. **הדרכה והכשרה** - הכשרת אנשים לכתיבת תכניות מהירה יותר.

מערכת הפעלה

מערכת הפעלה - Operating System - של המחשב כוללת אוסף של תוכניות שירות ותוכניות להידור שפות (Compilers).

מיכלול השרותים הזה מופעל בפיקוח של תוכנית בקרה (control program) הנמצאת כל הזמן בזכרון המחשב.

ההתפתחות הגדולה שחלה במהירות עיבוד הנתונים במחשב הותירה בפיגור את שיטות העבודה שהיו קיימות. זמנו של המחשב המשוכלל יותר הוא יקר יותר, ולאור זאת פותחו שיטות עבודה חדשות שנועדו לצמצם, אם לא לבטל כליל, את הזמן שבו היתה מערכת המחשב כולה, או אחד ממרכיביה, בלתי מנוצלים.

מערכת ההפעלה מיועדת להשגת מטרות עיקריות אלו:

1. הגדלת הספק המערכת, שהוא סה"כ כמות העבודה המתבצעת בפרק זמן נתון;
2. הקטנת זמן התגובה של המערכת, שהוא הזמן החולף בין הגשת עבודה לבצוע לבין מועד קבלת תוצאות העיבוד.
3. שליטה על יחידות האחסון במערכת לאבטחת התוכניות והקבצים מפני פגיעה.

בנוסף לכך נותנת המערכת שרותים להקלת העבודה ושיפור השליטה של מנהל יחידת המחשב, התוכניתן ומפעיל המחשב.

שרותים של מערכת ההפעלה

החלק המרכזי במערכת ההפעלה הוא תוכנית הבקרה (control program) אשר נטענת למחשב במכוון על ידי המפעיל כאשר הוא מתחיל את פעולת המחשב. התוכנית נטענת מספרית התוכניות של מערכת ההפעלה הנמצאת בדיסק מגנטי. מכאן ואילך מועבר ניהול פעולת המחשב אל תוכנית הבקרה אשר דואגת להפעלה סדירה של המחשב אולם נמצאת כל הזמן בשליטת המפעיל - האדם. שליטה זו מתבצעת באמצעות לוח הבקרה ומכונת הכתיבה הקשורה למחשב, אך בעיקר באמצעות הוראות הפיקוח של העבודות השונות (Job control). כאשר תוכנית הבקרה פועלת היא טוענת לזכרון תוכניות שרות שונות אשר נדרשות במהלך בצוע פעולות הבקרה והשליטה או עבודות המשתמש.

אחד התפקידים החשובים של מערכת ההפעלה הוא הפיקוח על זרם הפעלה רצופה של העבודות. כלומר, הזנת עבודות חדשות לבצוע באופן אוטומטי מקורא הכרטיסים ללא התערבות המפעיל. הוראות הפיקוח המוזנות יחד עם התוכניות והנתונים מאפשרות לתוכנית הבקרה ל"דעת" מה נדרש לעשות: לבצע הידור של תוכנית, לבצע הידור של תוכנית ולהפעילה מיד עם נתונים המצורפים לה או נמצאים בסרט או דיסק מגנטי, לבצע טעינה של תוכנית מספרית תוכניות ולהפעילה עם נתונים כלשהם וכדי.

מערכת ההפעלה מודיעה למפעיל, (באמצעות מכונת הכתיבה הצמודה למחשב, על העבודה המתבצעת בהתאם להנחיות הוראות הפיקוח). היא מאפשרת לו, במקרים מסוי-מים, להגיב או להזין פרמטרים הנדרשים לבצוע תוכנית מסוימת, לבטל תוכנית ועוד. כאשר צריך להרכיב או להסיר סרטים מגנטיים או דיסקים מגנטיים מקבל המפעיל הודעה מתאימה.

שרות חשוב נוסף של מערכת ההפעלה הוא פיקוח ותאום פעולות קלט/פלט. העברת הנתונים בין יחידות קלט/פלט לבין זכרון המחשב כרוכה בפעולת תאום. תחי-לתה ביצירת הקשר אל יחידת קלט/פלט כאשר מופיעה בתוכנית הוראה לקריאה או כתי-בה. אם הקשר נוצר (אלקטרונית) יכולים הנתונים לעבור. כאן יש לפקח על כמות הנתונים שעוברים בהתאם להוראות התוכנית ועל בדיקת אמינות ההעברה - כלומר, אם יש שגיאות אם לאו.

שרות אחר של מערכת ההפעלה מכוון להפעלה של תוכניות אחדות במקביל. מכיון שתוכנית אחת אינה מנצלת בדרך כלל את כל משאבי המחשב שהם הזכרון, יחידת העיבוד וכל יחידות קלט/פלט, אפשר להפעילם לסרוגין לשרותם של תוכניות אחדות. פעולה זו נעשית ע"י הטענת תוכניות אחדות לזכרון והפעלתן. הקצאת השרותים לכל תוכנית לפי סדר עדיפות נעשה ע"י מערכת ההפעלה. האפשרות לפעולה מקבילה של תוכניות קיימת מכיון שתוכנית מפסיקה עיבוד מדי פעם כדי לבצע פעולות קלט או פלט מיחי-דה מסוימת. כאשר היא משחררת את יחידת העיבוד, מנצלת אותה תוכנית אחרת אשר באותו זמן משחררת את מנגנון הטיפול ביחידות קלט פלט, וכן הלאה.

שרות נוסף הוא השמירה על קבצי נתונים וספריות תוכניות מפני כתיבה לא מכוונת אשר יכולה לפגוע בהם. פעולה זו נעשית באמצעות "סיסמאות" שאינן אלא הוראות מיוחדות של התוכנית או של מפעיל המחשב המתירים התחלת פעולת כתיבה בקו-בץ נדרש.

הפעלת תוכנית בפיקוח מערכת ההפעלה

לשם הפעלת התוכנית צריך לבצע שתי פעולות עיקריות:

א. הידור (Compilation), כלומר תרגום תוכנית המקור לתוכנית בת-בצוע. לכל שפת תכנות יש מהדיר המיוחד לה.

ב. הטענת התוכנית והפעלתה עם נתוני הקלט.

בזמן נפוי תוכניות מבצעים הידור והפעלה של התוכנית ברצף אחד. לכן יש לכלול בחבילת הכרטיסים המוגשת למחשב את תוכנית המקור ואת נתוני הקלט לתוכנית. כדי להורות למחשב (יותר נכון: למערכת ההפעלה הנמצאת בו) מה ואיך רוצים להפעיל עם חבילת הכרטיסים יש להוסיף לכרטיסים אלו גם הוראות הפעלה.

הוראות ההפעלה שונות לכל מחשב והן תואמות למערכת ההפעלה הנמצאת בו. תאור כללי לחבילת הכרטיסים שיש להגיש לבצוע הרצה במחשב:

1. כרטיס הוראת בצוע עבודה כולל את שם העבודה, פרמטרים להגבלת זמן בצוע וכמות נייר שתודפס ופרמטרים תמחיריים	card	JOB	
2. כרטיס פיקוח להפעלת המהדיר	card	EXEC	COMPILER
3. כרטיסי מקור של התוכנית כפי שנוקבו מתוך טפסי התוכנית	cards	PROGRAM	SOURCE
4. כרטיס סיום סטנדרטי לקובץ כרטיסים	card	LAST	standard
5. כרטיס פיקוח להפעלת התוכנית לאחר שעברה תהליך הידור	card	EXEC	PROGRAM
6. כרטיסי נתונים המשמשים קלט להפעלת התוכנית	cards	DATA	INPUT
7. כרטיס סיום סטנדרטי לקובץ הכרטיסים	card	LAST	standard



האם זהו כל הסוד !

חלק שני:

אלגוריתמים
ותרשימי עיבוד

פרק ו: תרשימים ואלגוריתמים

מהו תרשים

תוכנית עבודה למחשב (אשר נכתבת בשפת תכנות כלשהיא) מורכבת מסידרה ארוכה של פעולות מאורגנות, בסדר הגיוני מסויים. תכניות כאלו לרב ארוכות ומורכבות, ולעיתים המעבר מפעולה לפעולה אינו ישיר, או קבוע, ולכן גם מספר ההסתעפויות בתכנית הוא רב.

שומה על כותב תכנית שכזו לתאר את עבודתו ב"ראשי פרקים", כי התאור המילולי, אשר תלוי בכושר ההתבטאות של הכותב - יכול להיות מסורבל ולא ברור די צרכו.

"ראשי פרקים" אלה יירשמו בתוך צורות הנדסיות שונות (מלבן, מעויין, מקבילית וכו') אשר יחוברו ע"י חיצים. צורת הצגה זו נקראת תרשים עבודה.

מבחינים בין שני סוגי תרשים עבודה:

1) תרשים זרימה (FLOW CHART)

2) תרשים מלבני (BLOCK DIAGRAM)

תאור אלגוריתמים

תרשימי הזרימה והתרשימים המלבניים מכוונים בדרך כלל, בהקשר שלפנינו, לתאר תהליכי עיבוד נתונים. השמוש בתרשים ומדת הפרוט שלו מכוונת למטרה שאותה רוצה להפיק מכין התרשים.

בנוסף לתאור תהליכי עיבוד נתונים אפשר לתאר בעזרת תרשימים גם אלגוריתמים - תהליכים לוגיים לפתרון בעיה כלשהיא. סדרת הצעדים הדרושים לפתרון מפורטים בתבניות התרשים השונות.

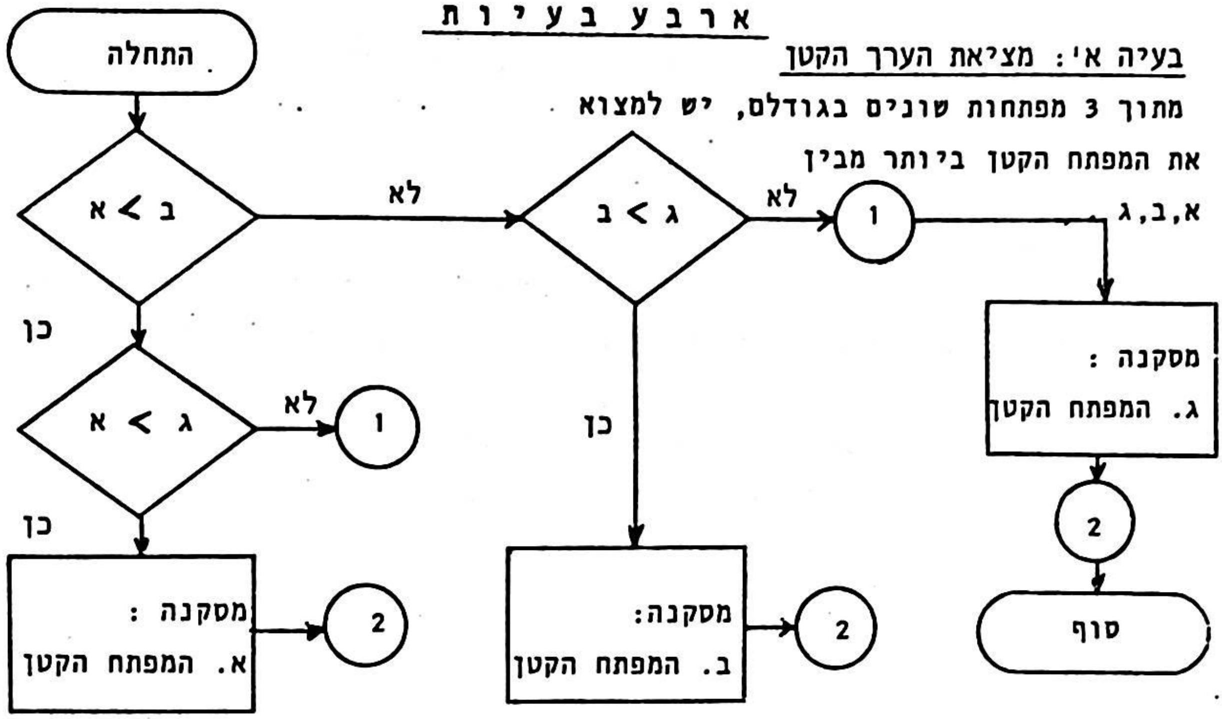
להלן מספר דוגמאות שאינן מתחום עיבוד הנתונים.

השתדל לעקוב אחר התהליך, על משמעות התבניות השונות נעמוד בפרוט בפרק הבא

ארבע בעיות

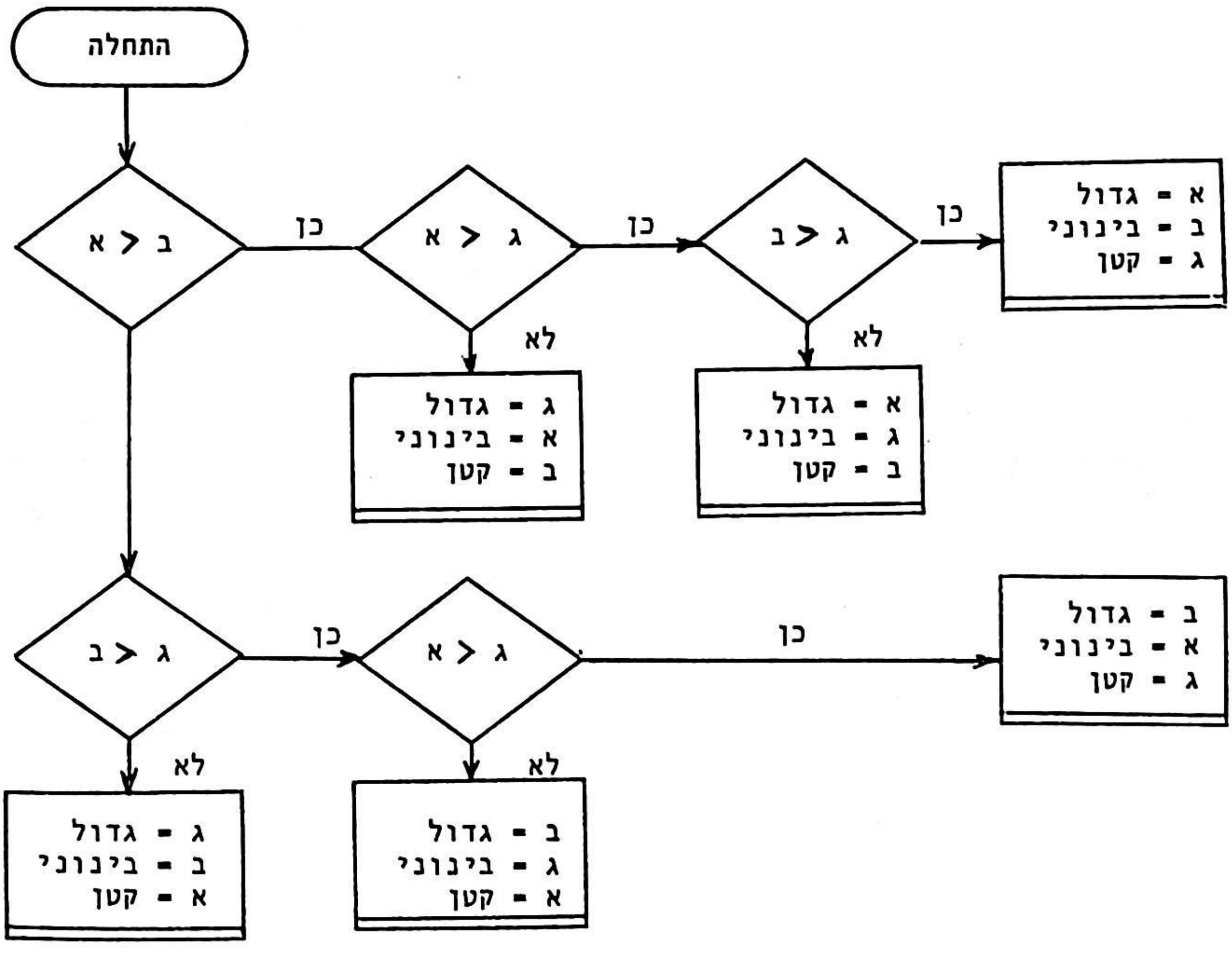
בעיה א': מציאת הערך הקטן

מתוך 3 מפתחות שונים בגודלם, יש למצוא את המפתח הקטן ביותר מבין א,ב,ג.



בעיה ב': קביעת יחסים בין 3 ערכים

תכנית לאפשרויות השונות של היחסים בין הערכים א,ב,ג.
עם מספר החלטות מינימלי .



<u>בעיה ג׳-נשים קנאיות</u>

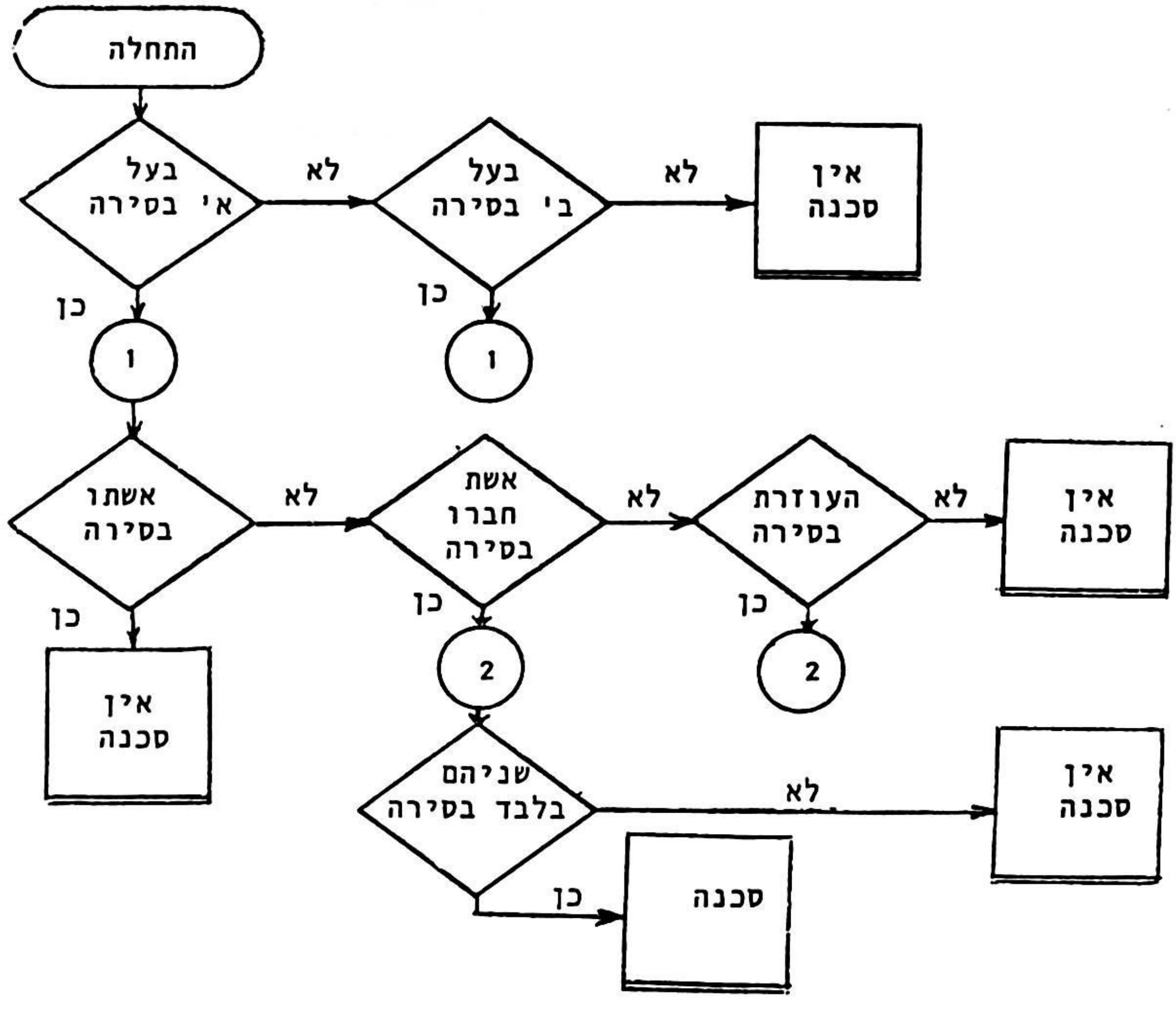
קיץ אחד בילו שתי משפחות את חופשתן השנתית על שפת האגם והשתכנו בבתים סמוכים הנשים, שהיו שתיהן קנאיות, החליטו שאסור שאחד מהבעלים יצא לשיט באגם עם אשת חברו בלי העוזרת.

מאחר שהסתבר שהעוזרת היא בחורה נאה, החליטו השתיים שגם עמה בלבד אסור לאחד מהבעלים לצאת לשיט.

חשמלאי התקין מערכת איתות חשמלית, אשר פעלה בין בית הסירות וחדרי המגורים של שני הבתים. בכל אחד מחדרי המגורים היו מנורות בשני צבעים - אדום וירוק. האדום סימן סכנהו ויידלק ברגע שאחד מהבעלים יפר את ההסכם. הירוק יידלק בכל שאר המקרים - היינו, אין סכנה.

שתי הנשים הטילו על האחראי לסירות להפעיל את האיתות בהתאם להסכם.

יש לתאר את אפשרויות היציאה לשיט בתרשים זרימה.



בעיה די - איך לקום בבוקר

מר אי מתעורר בבוקר עם צלצול השעון המעורר.

הוא מתמתח ומנסה להעיר את אשתו. חוזר 3 פעמים על שתי פעולותיו האחרונות.

יוצא ממיטתו ומסתכל דרך החלון. אם חושך בחוץ - מעלה אור בחדר.

הוא מעיף מבט במד-החום. כשיש פחות מ -20^0 אך יותר מ - 10^0, מדליק את התנור.

אם החום פחות מ - 10^0, חוזר למיטתו ובשלב זה מסיים את הנסיון לקום בבוקר שכזה.

כשמד-החום מראה 20^0 או יותר, הולך מר אי להתרחץ וניגש לאכול ארוחת-בוקר. אם זקוקה אשתו לכסף, נותן לה 20 לירות.

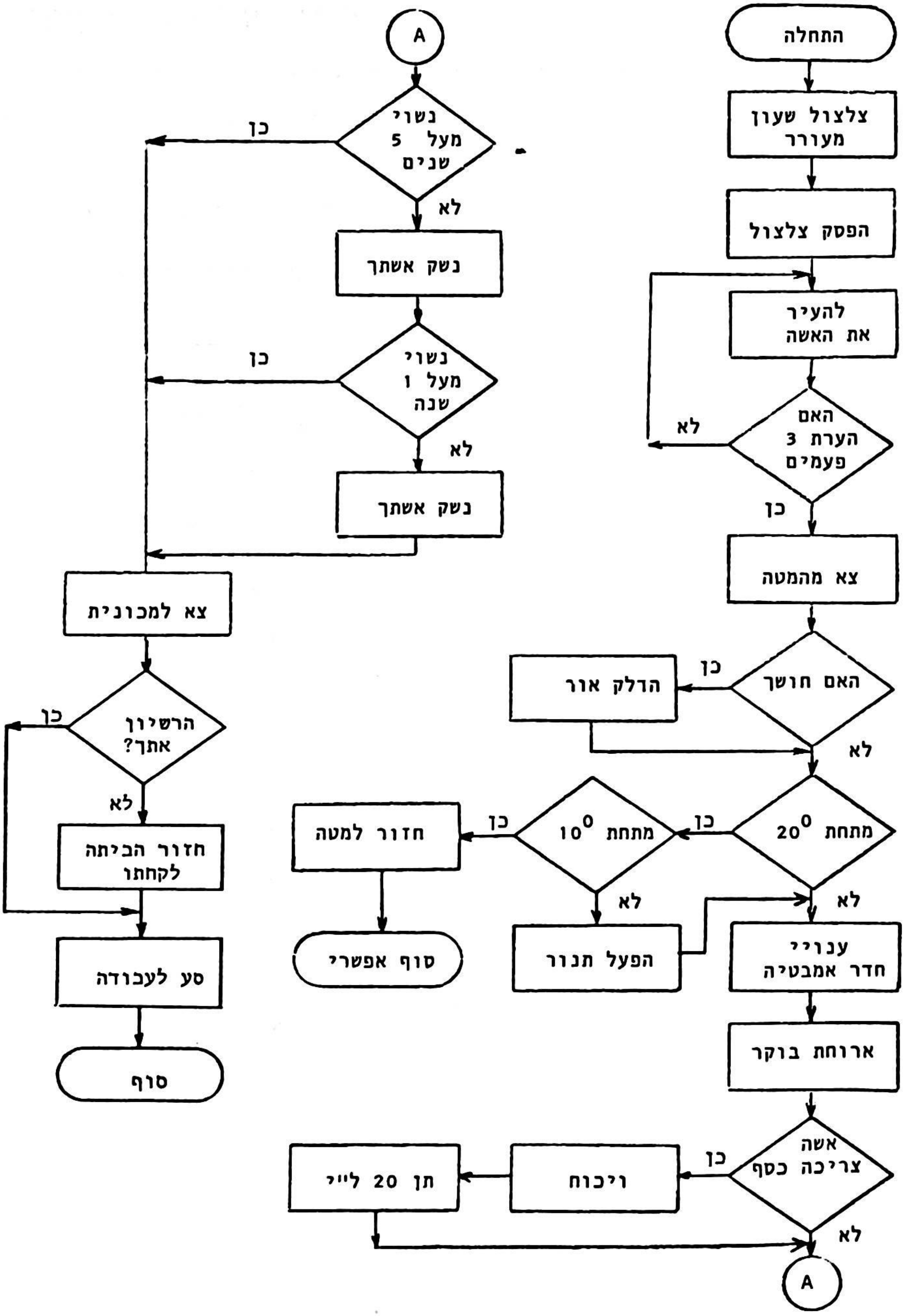
במקרה שהם נשואים יותר מחמש שנים - יוצא למכוניתו.

נשואים פחות מחמש שנים - נותן נשיקה לאשתו.

נשואים פחות משנה - נשיקה נוספת לאשתו.

בהכנסו למכונית בודק מר אי האם רשיון הנהגות נמצא אצלו. אם איננו-חוזר הביתה לקחתו. אם רשיון הנהגות בידו, נוסע מר אי לעבודה.

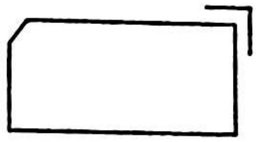
ז לתאר את תוכן הספור בעזרת תרשים.


התחלה
צלצול שעון מעורר
הפסק צלצול
להעיר את האשה
האם הערת 3 פעמים
לא
כן
צא מהמטה
האם חושך
כן
הדלק אור
לא
מתחת 20^0
כן
מתחת 10^0
כן
חזור למטה
סוף אפשרי
לא
הפעל תנור
לא
עניי חדר אמבטיה
ארוחת בוקר
אשה צריכה כסף
כן
ויכוח
תן 20 לייי
לא
A
A
נשוי מעל 5 שנים
כן
לא
נשק אשתך
נשוי מעל 1 שנה
כן
לא
נשק אשתך
צא למכונית
הרשיון אתך?
כן
לא
חזור הביתה לקחתו
סע לעבודה
סוף

פרק 2: תרשימי זרימה

תרשים זרימה הינו תאור גרפי סכימטי של תהליך הכולל: הקלט לסוגיו השונים, שלבי העיבוד אותם הוא עובר במחשב, ולבסוף - הפלט.

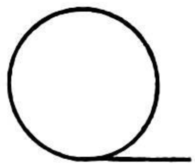
הסימנים הסטנדרטיים לתרשים זרימה הם:



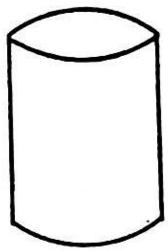
כרטיס מנוקב



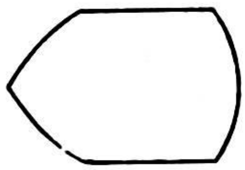
סרט נייר מנוקב



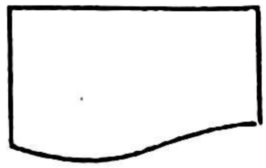
סרט מגנטי



דיסק מגנטי



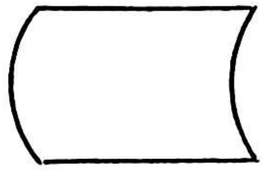
אמצעי להצגה חזותית (מסך)



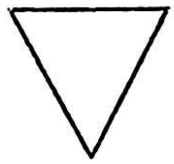
מסמך מודפס



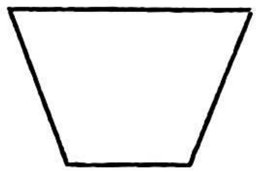
קלט ידני (לדוגמה: לוח מקשים)



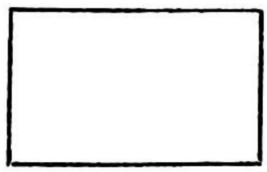
אמצעי אחסון מגנטי, המחובר למחשב (סרט מגנטי,
תוף מגנטי).



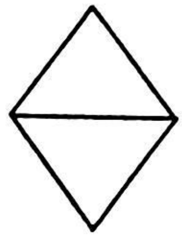
אמצעי אחסון חיצוני, אינו מחובר למחשב (כרטיס
מנוקב וכו').



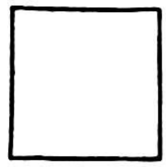
עיבוד ידני



עיבוד במחשב (מיון, חשובים, עריכת נתונים
השוואות וכדי).



מיון כרטיסים



פעולת עזר (לא תחת הפקוח של ה - CPU)



חץ המקשר את הצורות השונות.

הפעולה הנדרשת נרשמת <u>בתוך</u> הצורה ההנדסית.

בעיה א' - דו"ח

1. **תאור הבעיה**

א. **נתון**: קובץ מלאי בכרטיסים, כרטיס לכל פריט.

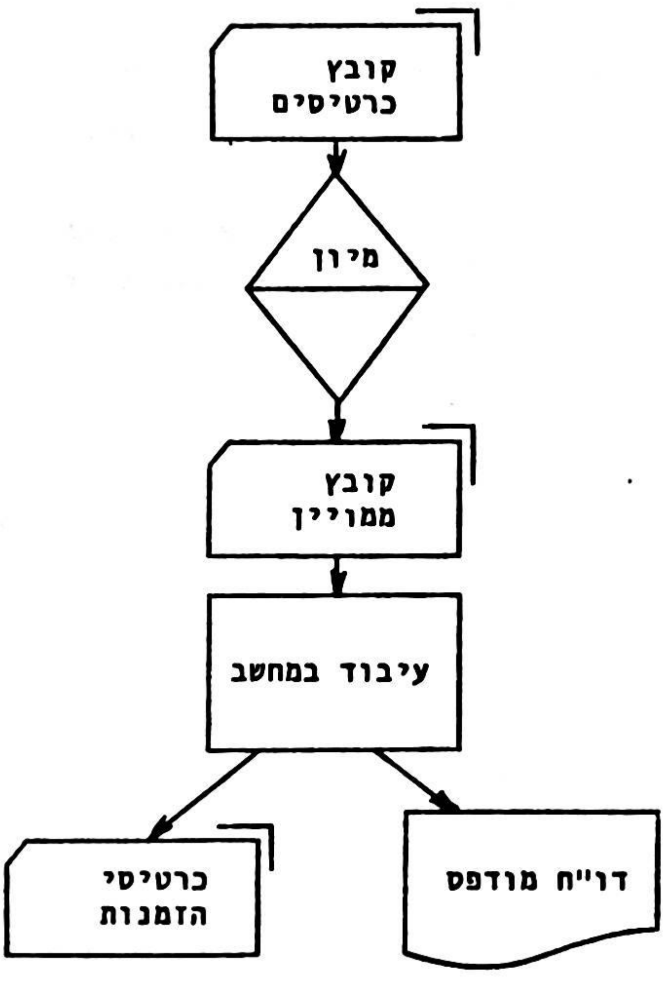
ב. **עיבוד**:

1) להפיק דו"ח מודפס לפי מספר קטלוגי עולה, עבור כל הפריטים שבמחסן.

2) לבדוק את רמת (כמות) המלאי ולהשוותה לרמת מינימום מותרת (שני שדות אלו נמצאים ברשומת הפריט בכרטיס). אם רמת המלאי שוה או נמוכה מרמת המינימום - יש לנקב כרטיס הזמנה עבור אותו פריט.

ג. הערה: הקובץ אינו ממויין לפי מספר קטלוגי של הפריט עקב תהליך עיבוד קודם.

2. **תרשים הזרימה**



3. **ניתוח הפתרון:**

בפתרון זה ניתן להבחין בקלט המקדים פעולה כלשהיא, ובפלט - שהינו תוצאה של תהליך העיבוד. בתהליך זה פלט של פעולה אחת (מיון) משמש כקלט לפעולה הבאה (עיבוד). כל שלב של פעולה יכול לקלוט יותר מקובץ - קלט אחד, וגם הפלט יכול להיות מגוון ומורכב מקבצים אחדים. הקלט יכול להיות קובץ באמצעי אחסון מסוג אחד (כרטיסים למשל) והפלט - קובץ באמצעי אחסון דומה או שונה (כרטיסים, תדפיס, סרט מגנטי וכדי).

בעיה בי - עדכון קובץ פריטים במחסן

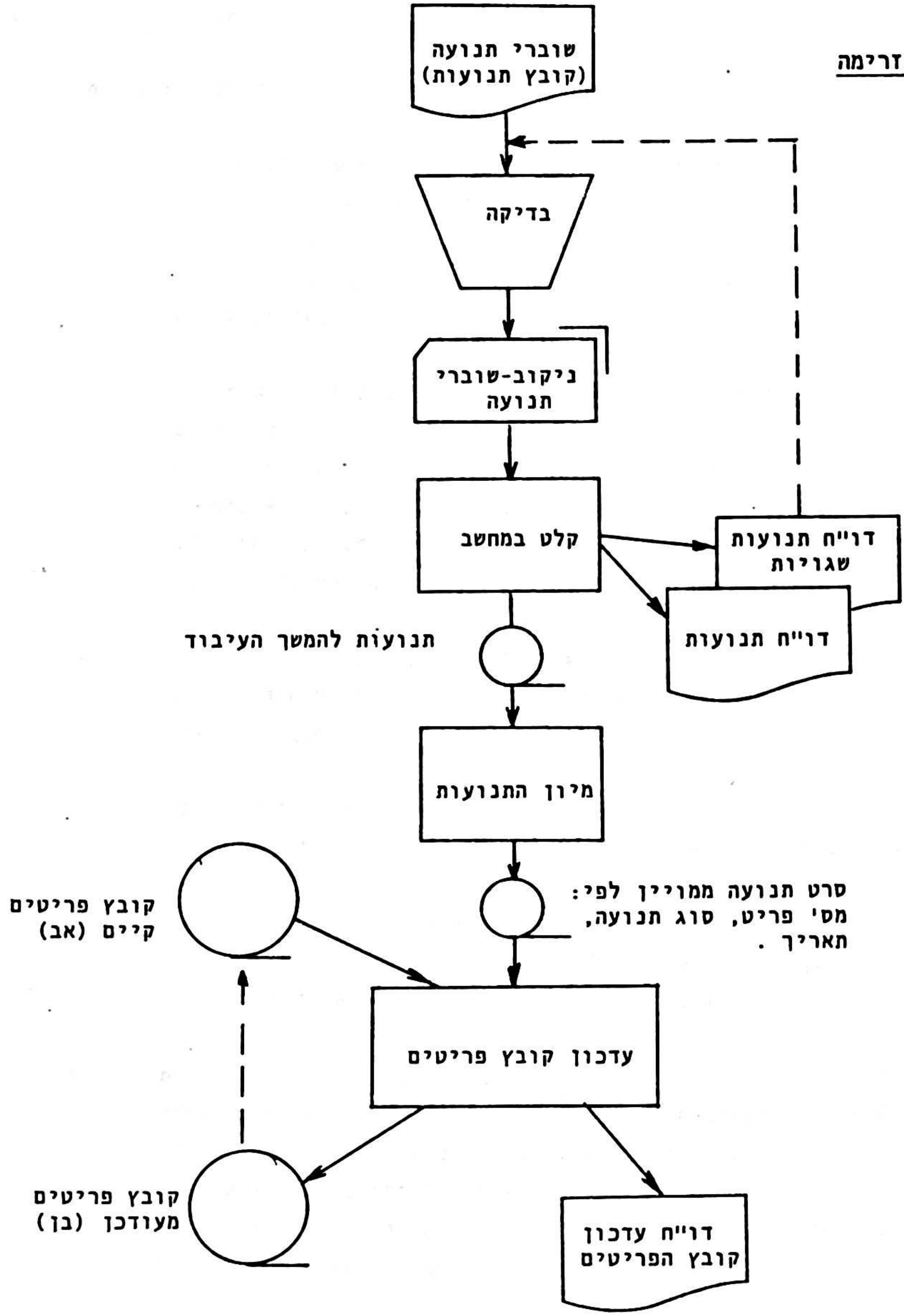
1. **תאור הבעיה**

א. **נתון:** **קובצי הקלט:**
קובץ אב - סרט מגנטי
קובץ תנועות - כרטיסים בלתי ממויינים

ב. **עיבוד:** יש לעדכן את קובץ האב לפי כרטיסי התנועות, לפני העדכון יש לבצע בדיקות לוגיות.

ג. **הפלט:** קובץ אב מעודכן
דו"ח מודפס של התנועות

2. תרשים הזרימה



3. <u>ניתוח הפתרון</u>

תרשים הזרימה לתאור היישום כולל מספר שלבים.

א. בדיקת הנתונים ממסמכי המקור. בבדיקה זו נבדקים פרטים טכניים שונים הקשורים ברישום הפרטים השונים במסמך כגון: אסמכתא, סוג פעולה, תאריך, מספר קטלוגי ועוד.
כמו כן נבדק, עד כמה שאפשר הרישום עצמו: למשל האם התאריך רשום בצורה נכונה, האם המספר הקטלוגי מכיל את כל הספרות הדרושות וכדומה.

ב. העברת הנתונים מטפסי המקור לקלט קריא במחשב, ע"י תהליך ניקוב בכרטיסים.
בדרך כלל מועברים הכרטיסים המנוקבים בתהליך בקורת. תהליך זה זהה לתהליך הניקוב אולם המכונה אינה מנקבת אלא משוה את פעולת ההקשה של המפעילה אל הניקוב בפועל בכרטיס. בכל מקרה של אי התאמה ניתנת התראה והמפעילה יכולה לתקן את הכרטיס השגוי.

ג. קליטת התנועות במחשב ובדיקות לוגיות של הנתונים: בדיקת חוקיות רישום נתונים שונים, בדיקות התאמה של מונים שונים בתנועה, בדיקת סימולים וכדי. הפלט של מעבר זה כולל:

1) דו"ח תנועה אשר מפרט את כל התנועות שהתקבלו ומציין את התנועות שהועברו להמשך העיבוד כקובץ תנועה ואת התנועות שנפסלו.

2) דו"ח תנועות שגויות אשר מפרט את התנועות השגויות ומציין במפורש את השגיאות שנתגלו. אנשי הביקורת יבצעו את התקונים הדרושים ויחזירו את התנועות האלו לתהליך העיבוד.

ד. מיון קובץ התנועות לקראת עדכון קובץ הפריט. המיון יבוצע לפי מפתח הרשומות בקובץ הפריט, במקרה זה - המספר הקטלוגי. בנוסף לכך יש למיין גם לפי סוג תנועה (ניפוק, קבלה, החזרה וכדי) ולפי תאריכים. מפתחות מיון אלו נדרשים כדי לקיים את הלוגיקה הנכונה של פעולות העדכון.
המיון כולו נעשה במעבר אחד - במפתח מיון אשר מורכב מכמה שדות: מס' קטלוגי (ובתוכו) סוג תנועה (ובתוכו) תאריך.

ח. למהלך העדכון מוזן קובץ התנועה הממויין וקובץ הפריטים הקיים.
ממהלך זה יופק קובץ פריטים מעודכן.
המושגים קובץ "אב" וקובץ "בן" מתיחסים לתהליך שבו מופק קובץ חדש מעודכן מקובץ קודם.
קובץ ה"בן" ישמש במהלך העדכון הבא כקובץ "אב" ליצירת קובץ "בן" חדש.
בתהליך העדכון מופק גם דו"ח עדכון אשר מפרט את כל הפעולות שנעשו בכל פריט ופריט. הדו"ח מציין את היתרה הקודמת והיתרה החדשה לשם בקורת הרישום ואפשרות בקורת במחסן. לדו"ח אפשר לצרף חריגים ושגיאות שנתגלו במהלך העדכון.

בעיה ג' - ניהול מלאי, חשבונות ספקים וחשבונות לקוחות

1. ניהול המלאי

א. קלט: קובץ התנועות וקובץ המלאי.

ב. עיבוד: עדכון הרשומות בקובץ המלאי אשר כולל שינוי, הוספה וביטול. חשובים שונים הנדרשים בעדכון.
בתהליך העדכון מתבצעות פעולות שונות שמתיחסות לכמויות הפריט במלאי וגם לנתונים אחרים תאוריים כגון: שם הפריט, מחיר, יחידת מדה, ספקים מורשים לפריט, נקודת הזמנה וכד'.

ג. פלט: קובץ מלאי מעודכן, דו"ח עדכון מלאי.
דו"ח המלאי מכיל לכל פריט נתונים על: יתרת הפתיחה, התנועות שבוצעו ויתרת הסגירה. הדו"ח יכול לפרט כמויות במלאי וערכים כספיים, מקרים חריגים כגון ירידה ממלאי מינימום, תנועת מלאי גדולה מערך נתון. ניתן גם לציין את הפריטים שלא חלה בהם תנועה במשך זמן רב (לפי תנאי הבעיה) לציון מלאי מת, ועוד.

. ניהול קובץ ספקים ולקוחות

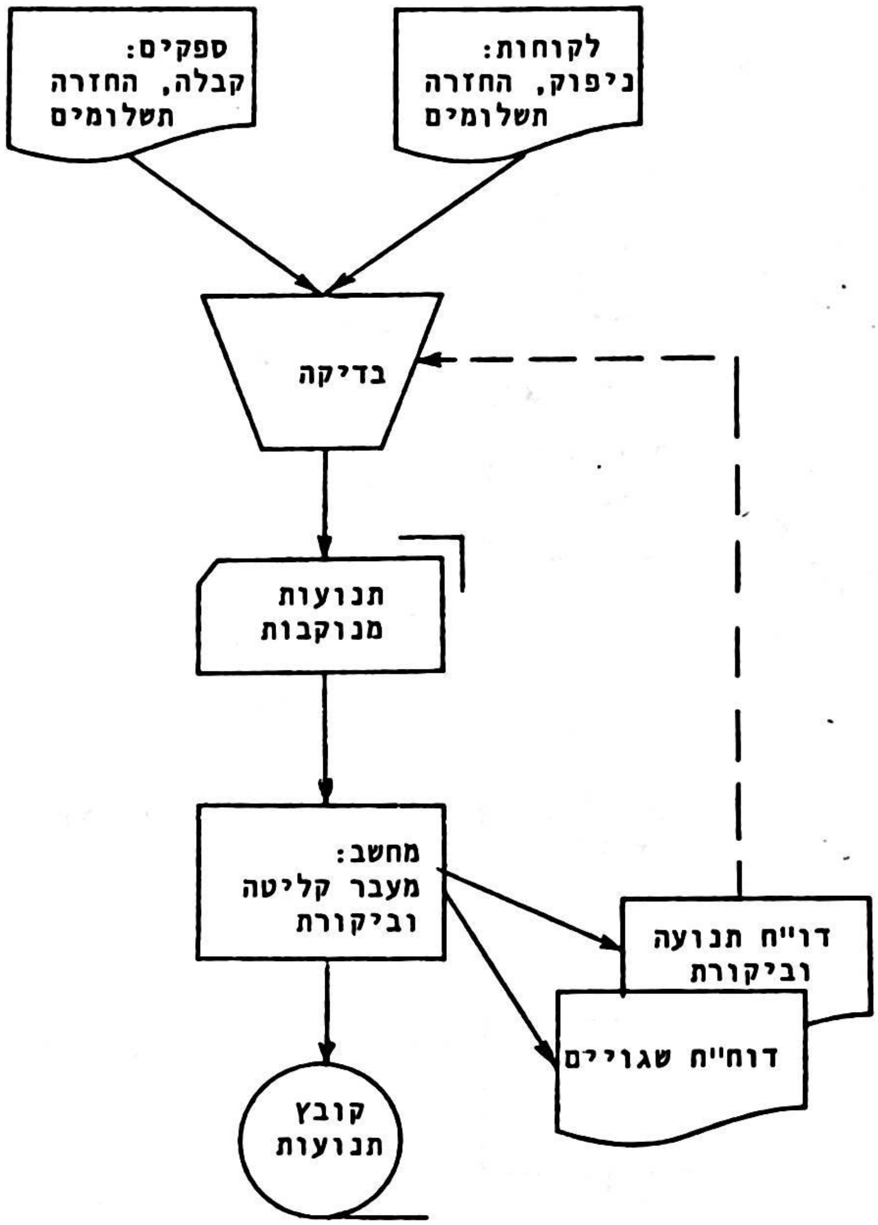
א. קלט: קובץ תנועות וקובץ ספקים ולקוחות.

ב. עיבוד: חישוב יתרת הזכוי/חיוב בהתאם לצורך והוצאת חשבונות. רכוז חשבונות.
בתהליכי עיבוד מסוימים אפשר להפיק גם הוראות תשלום ולעדכן בהתאם את חשבון הבנק אשר מנוהל במערכת.

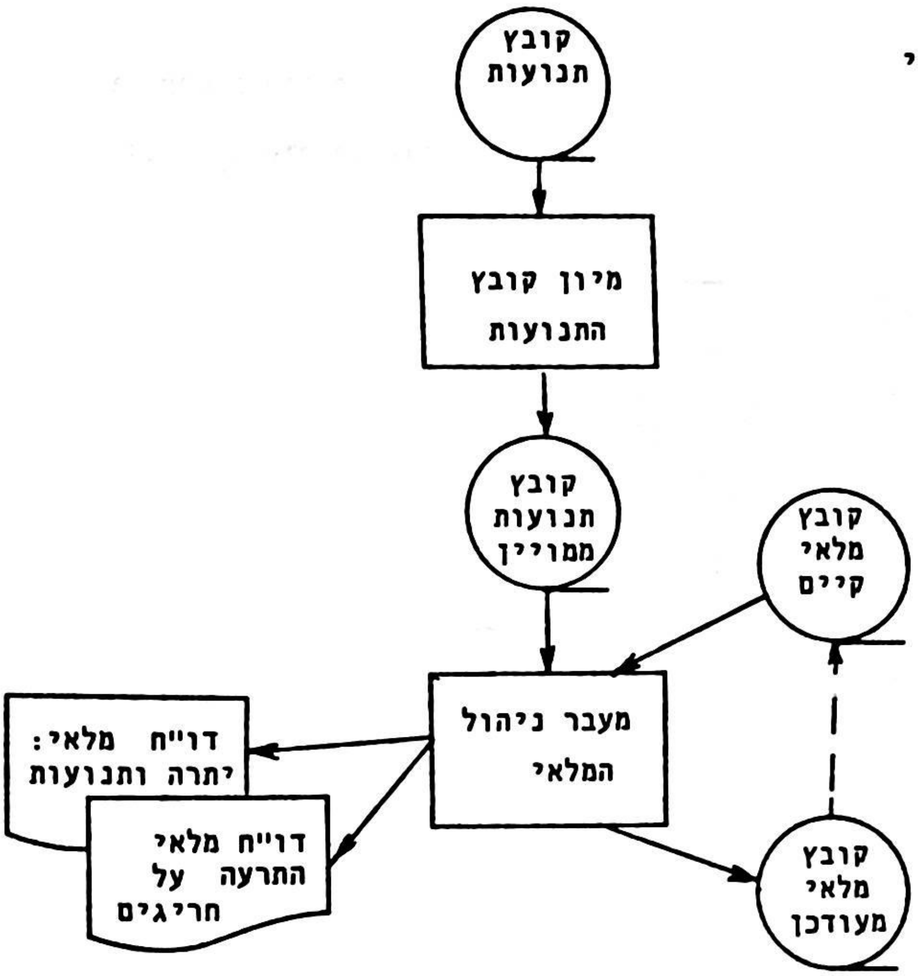
ג. פלט: קובץ ספקים ולקוחות מעודכן, דו"חות שונים.
הדו"חות המופקים יכללו רשימות של היתרות ההתחלתיות בחשבונות ספקים ולקחות עם פירוט התנועה והיתרה הסופית לכל חשבון.
דו"חות אחרים יספקו ריכוז של החשבונות ויציינו את היתרות הסופיות בחתכים לפי דרישת המשתמש.
בנוסף לכך יוכל המחשב להפיק חשבונות זכוי/חיוב אשר ישלחו לספקים או ללקוחות בהתאמה.

3. תרשים הזרימה

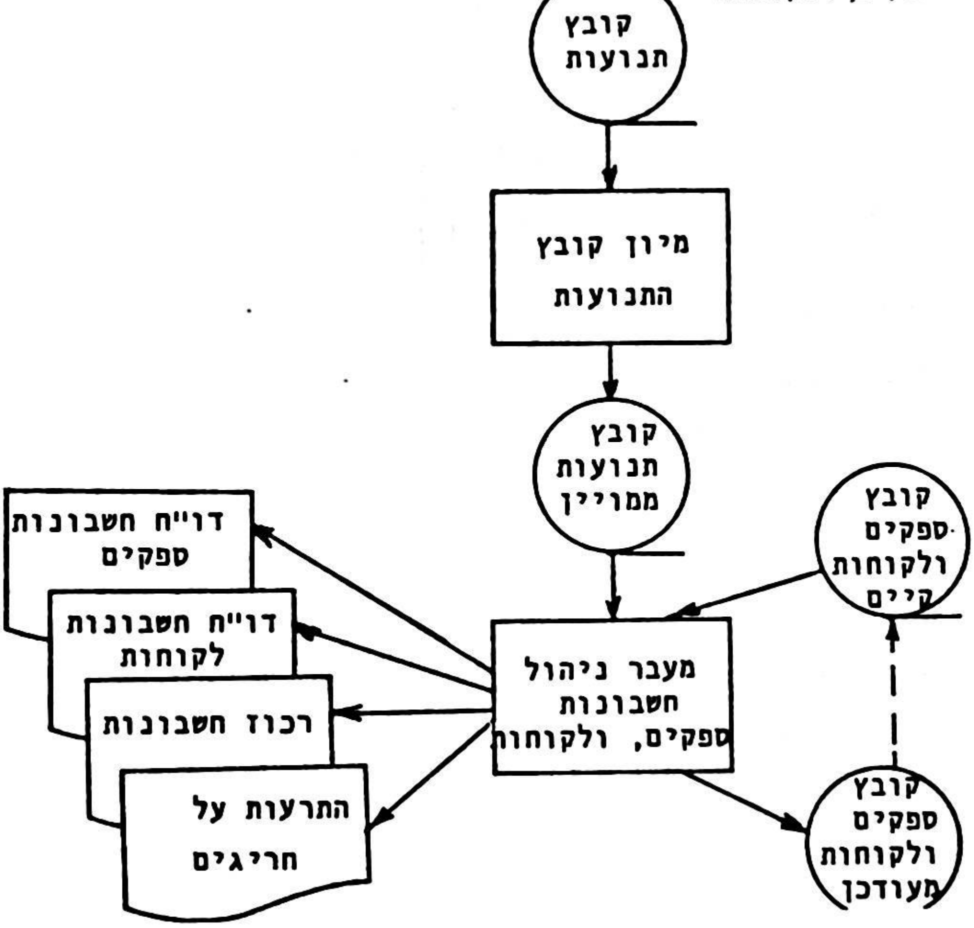
שלב א' : קלט תנועות למחשב



שלב בי : ניהול המלאי



ניהול חשבונות ספקים ולקוחות



4. נתוח הפתרון:

א. תרשים הזרימה לתאור היישום כולל מספר שלבים:
שלב א' - קליטת התנועות במחשב.
שלב ב' - ניהול קובץ המלאי.
שלב ג' - ניהול קובץ ספקים ולקוחות.

ב. השלב הראשון מכיל את כל הפעולות הדרושות להעברת מסמכי המקור השונים אל קלט קריא במחשב, לבדיקת התנועות והפקת קובץ תנועות "טובות". כל התנועות השגויות תרשמנה בדו"ח השגויים אשר יועבר לבדיקה, לתקון וחזרו לתהליך הקלט.

ג. שלבים ב' וג' מתחילים בפעולות מיון. בקובץ התנועות כלולות תנועות שונות אשר חלקן מיועדות לעדכון קובץ המלאי בלבד וחלקן-לעדכון קובץ הספקים והלקוחות בלבד,וחלק מהתנועות יכולות לפעול על שני הקבצים. מכיוון שכל התנועות כלולות בקובץ אחד יש למיין את אותו הקובץ פעמיים, עבור על אחד ממהלכי העדכון השונים - ולפי סדר המיון הנדרש ע"י הקובץ שאותו מעדכנים.
המיון הראשי יהיה - מספר הקובץ. בתוכו לפי המיון הראשי של הקובץ ובתוכו לפי שדות נוספים בהתאם לנוהל העדכון.

מיון התנועות לעדכון קובץ המלאי יהיה לפי מספר פ"יט בתוכו תאריך ובתוכו סוג תנועה (קבלה, ניפוק וכד').

הערה: בבעיה קודמת בצענו את המיון לפי מספר פריט, סוג תנועה ובתוכו תאריך, כפי שנדרש באותה בעיה.

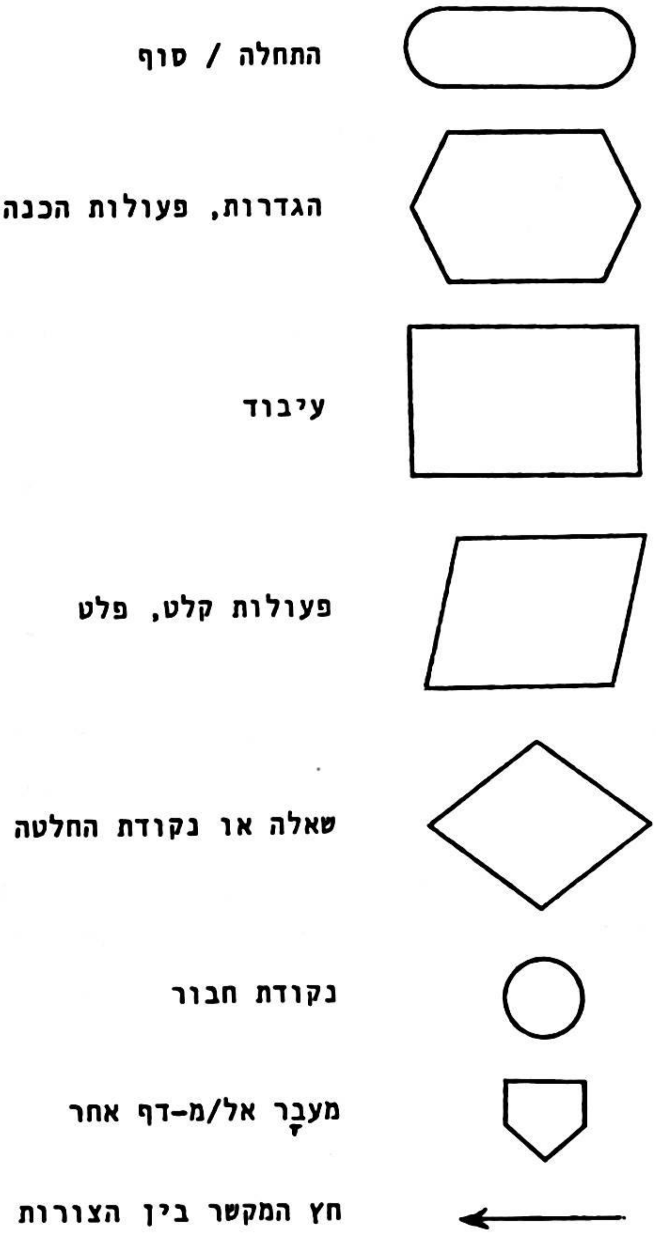
מיון התנועות לעדכון קנבץ ספקים ולקוחות יהיה לפי מספר ספק/לקוח, בתוכו תאריך ובתוכו סוג הפעולה (תשלום, חיוב וכו') . אם ישנם ברשומה שדות נוספים החשובים לסדר העדכון יש לכלול גם אותם במיון.

פרק 3: תרשים מלבני

תאור סכמטי של עיבוד הנתונים, המוצג ע״י תרשים הזרימה, מספיק עבור הדרגים המנהליים, שעיסוקם אינו קשור ישירות לעיבוד נתונים. מובן מאליו שאין תרשים כזה מספק את המתכנת, שתפקידו לכתוב תכנית מפורטת, לפיה יבצע המחשב את העבודה.

תרשים מפורט של פעולות המחשב נקרא <u>תרשים מלבני</u>.

הסימנים המקובלים לתרשים המלבני הם:



בעיה א' - הדפסת רשומות בקובץ

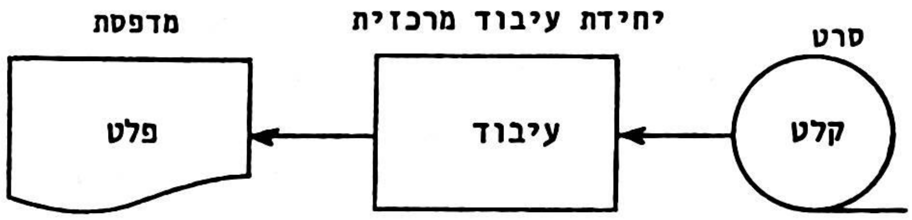
1. תאור הבעיה

א. נתון: קובץ סטודנטים בסרט מגנטי. לכל סטודנט יש רשומה.

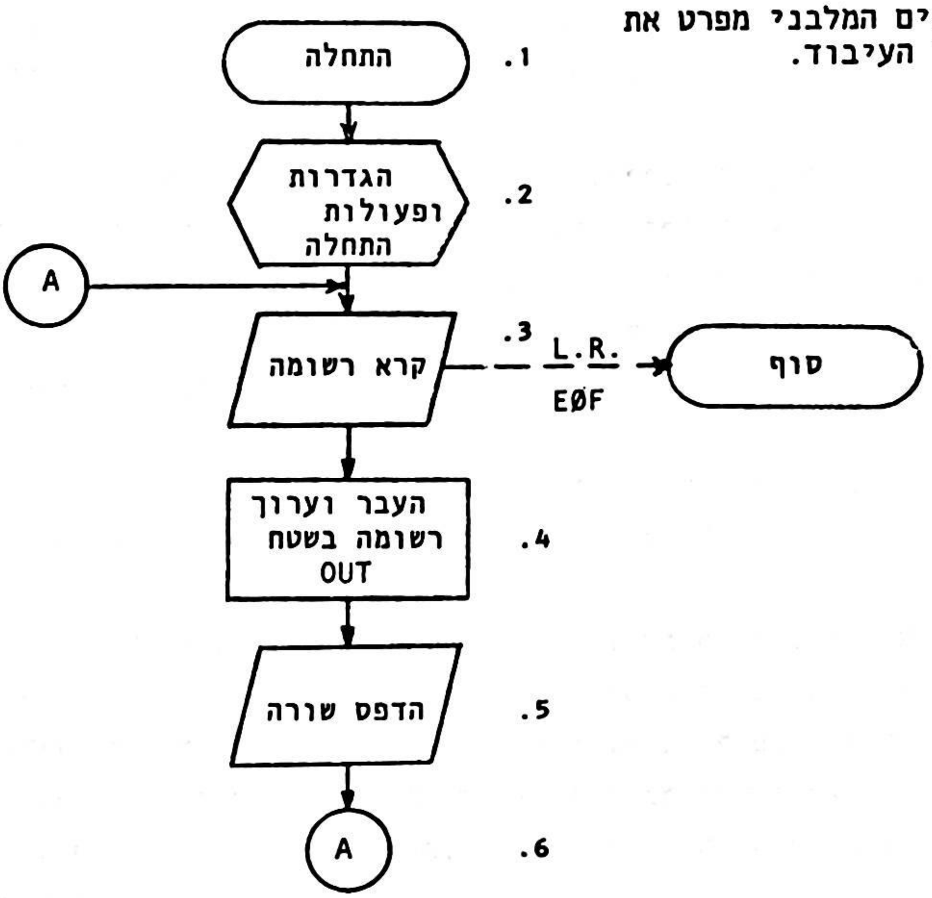
ב. פלט: דו"ח המכיל את רשימת כל הסטודנטים, כל אחד בשורה נפרדת.

ג. עיבוד: הדפסה. יש לעצור בסיום פעולת ההדפסה.

2. תרשים הזרימה לבעיה



3. התרשים המלבני מפרט את שלבי העיבוד.



4. נתוח הפתרון

א. התחלה - כל תרשים מלבני מתחיל בסימן זה. חץ מקשר אותו אל השלב הבא.

ב. הגדרות - בהגדרות נוהגים לציין שני סוגי הגדרות: הסוג הראשון מתיחס לסביבת העיבוד (ENVIRONMENT). כאן יש לציין אילו יחידות קלט ופלט ישותפו בעיבוד זה. (היות וכלל זה אינו ישים לגבי כל שפות התיכנות - בתרשים המלבני לא ייכתב דבר, פרט למלה "הגדרות").

הסוג השני מתייחס אל השטחים (שדות) במחשב, להם זקוק עיבוד ספציפי זה. בדוגמה שלפנינו הראינו שני שדות כאלו בשם IN ו - OUT .
בקבוצת פעולות זו כלולו גם "פעולות התחלה" (initialization) שונות. פעולות ההתחלה המקובלות הן איפוס מונים או הקניית ערכים התחלתיים למשתנים, הדפסת כותרות ופעולות שונות אחרות הנדרשות לפי תנאי הבעיה. בדרך כלל פעולות אלו מתבצעות פעם אחת בלבד בתחילת התוכנית, ואין חוזרים עליהן.

ג. קרא רשומה - זוהי פעולת קלט, (ולכן היא מסומנת בתוך מקבילית) בה נעשית העברת הנתונים מכונן הסרטים אל הזכרון הפנימי במחשב. עם ביצוע פעולה זו מועבר תוכן הרשומה הראשונה לשדה בשם IN.

ד. העבר וערוך רשומה בשטח OUT - פעולה זו(המצויינת בתוך מלבן) גורמת להעברת הנתונים לשדה בשם OUT. פעולה זו - הכוללת עריכת השורה בתבנית השורה המודפסת - הכרחית במקרה שלפנינו כי הנתונים נכתבו על הסרט ללא רווחים בצורה צפופה ולוא היינו מדפיסים אותם, כפי שהם מצויים על הסרט - ספק אם יכולנו להבין את הכתוב.

ה. הדפס שורה - זוהי פעולת פלט (שוב מקבילית) הגורמת להעברת הנתונים אל המדפסת, קידום הניר לשורה הבאה והדפסת השורה על הניר.

ו. סיעוף לנקודה A - בשלב זה של העיבוד בצענו את כל הפעולות הדרושות לקריאת רשומה אחת מן הסרט והדפסתה על ניר המדפסת. ברור, שלגבי הרשומות הבאות עלינו לחזור על הפעולות הללו. יותר נכון על פעולות 5 - 3.
נכַנה את פעולת הקריאה כפעולה מס' A, ואזי נוכל למשוך חץ מ"הדפס שורה" אל עגול ובתוכו נרשום את הסימן A, שהוא נקודת חבור עם העגול A, ליד פעולת הקריאה.

מעתה תחזור המערכת על בצוע הפעולות 5,4,3 בהתאם לחיצים המקשרים את הפעולות הללו ביניהן: תקרא רשומה, תערוך אותה כפלט, תדפיס אותה על המדפסת וחוזר חלילה.

עד מתי נהיה מעונינים בבצוע במחזור של פעולות 5,4,3 ? כמובן, עד סוף הקובץ, עד שנשלים את הפעולות המתיחסות לרשומה האחרונה שבקובץ.

על הסרט המגנטי נמצא סימן מיוחד, המציין את סוף הקובץ והמערכת תבחין בכך תוך כדי קריאת הסרט. יש לציין ליד פעולת הקריאה את הפעולות הדרושות לבצוע במצב של זהוי "סוף" הקובץ. במקרה שלנו - לעצור.

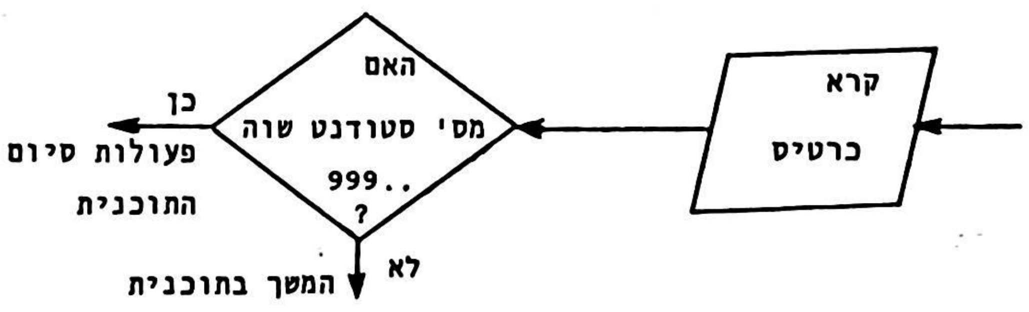
ז. סוף - פעולה זו נמצאת מימין להוראת הקריאה, ומחוברת עם חץ "מקווקו" אל התחנה האחרונה.לעתים נבדק סוף הקובץ במפורש ע"י שאלה .
מעל החץ נוהגים לציין EOF (סוף קובץ - End Of File) או (רשומה אחרונה - Last Record) .

תרשימים לבצוע פעולות חישוב פשוטות

בדוגמאות הבאות נניח תהליכי חישוב פשוטים אשר מהוים קטע ממערך תוכנית מורכב יותר. הבעיות שתוצגנה מיועדות להדגים את הבניה הלוגית של תהליך העיבוד.

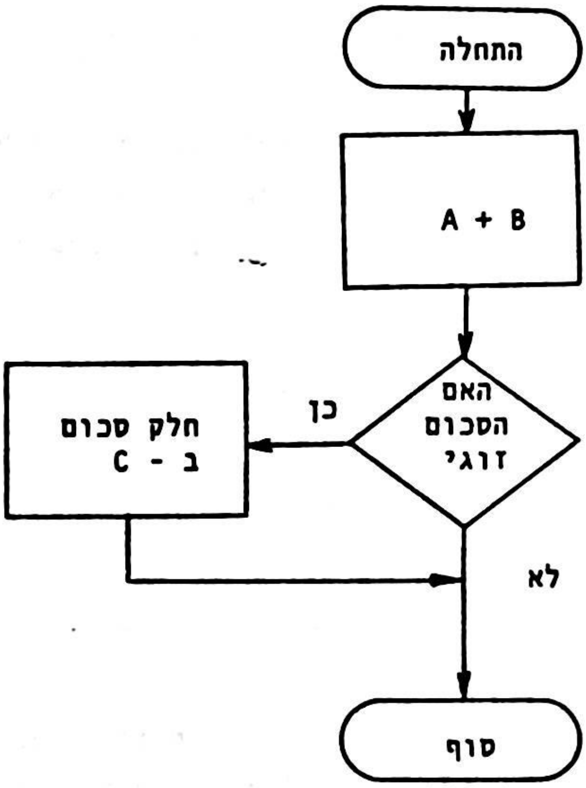
בפעולות קריאה של רשומות הנחנו שהתוכנית הסטנדרטית של המחשב המפקחת על בצוע פעולות הקריאה מזהה את הרשומה האחרונה (במבנה סטנדרטי) המציינת את סוף הקובץ ואשר אינה מכילה נתונים כלשהם. לפיכך, בצענו סיעוף מתוך הוראת הקריאה.

כאשר הכרטיס האחרון בקובץ אינו במבנה סטנדרטי של המערכת יש לבדוק אותו במפורש. נניח למשל שבכרטסת סטודנטים מכיל הכרטיס האחרון את המספר 999999 בשדה מספר סטודנט. התרשים יראה כך:



<u>בעיה ב׳ - חבור שני מספרים והתניה</u>

יש לחבר שני מספרים A ו- B . אם הסכום אינו זוגי, סיים את העבודה; אם הסכום הוא זוגי, חלק את הסכום ב - C וסיים.



התרשים הנכון הוא תרשים המטפל בכל תנאי הבעיה. כאשר ניסוח התנאים והפעולות מדויק ובהיר ומספר הצעדים הנדרשים הוא הקטן ביותר, אפשר גם להגדירו כתרשים טוב.

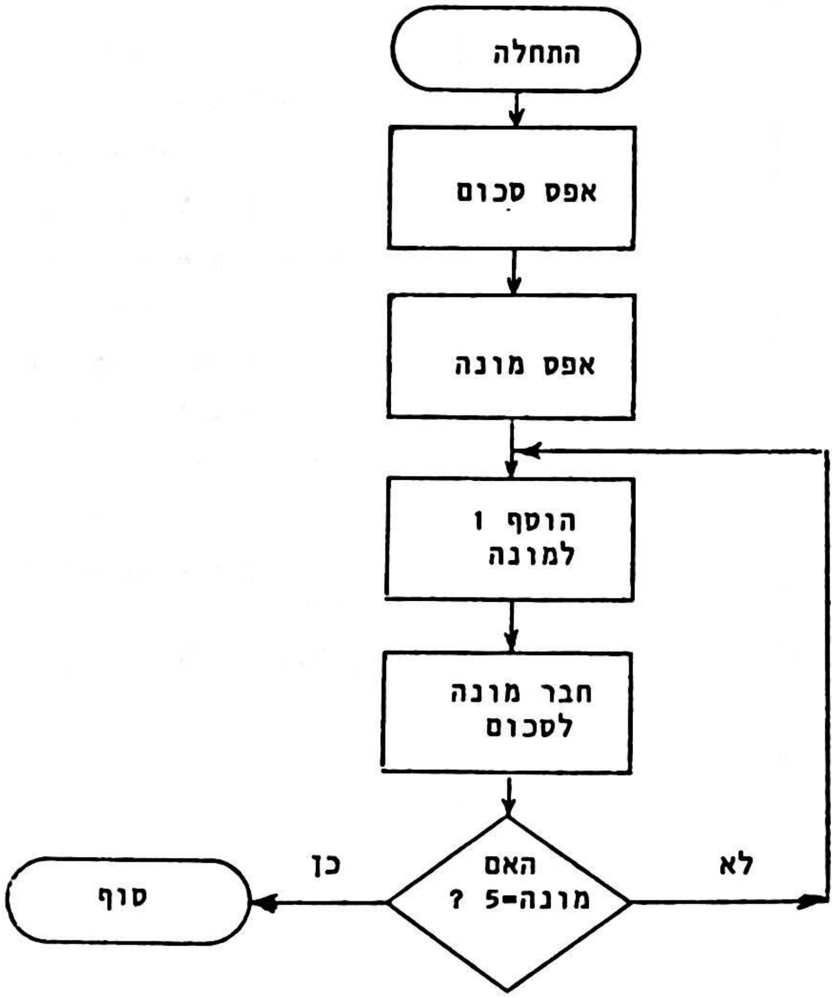
בעיה ג׳ -סכום סדרת מספרים בלולאה

1. **תאור הבעיה**

יש לסכם את כל המספרים מ - 1 עד 5. פעולה זו ׳ פעולות חבור של הערכים
1 עד 5 בהתאמה. כדי לשלוט על מספר הפעולות ... ׳הר זאת בעזרת מונה,
אשר יורה על סיום כאשר ערכו יגיע ל - 5 .

כאשר מסכמים למונה יש לאפס אותו לפני תחילת הפ׳ ׳ איפוס מונה נהוגה
במכונות סיכום פשוטות וכן יש לנהוג גם במחשב.

2. **התרשים המלבני**



המונחים "מונה" ו"סכום" מציינים את הפעולה הנדרשת על ידנו. למעשה בשני המקרים מבוצעת פעולת סיכום למונה, או בשפת המחשב - **פעולת חיבור למשתנה**.

3. <u>בדיקת התרשים</u>

כדי לבדוק אם התרשים עונה על תנאי הבעיה יש לבדוק אותו בצורה מסודרת בעזרת נתוני ניסוי.
בבעיה שלפנינו אין נתוני קלט וע"כ נתחיל מיד בתרשים. לצורך הבדיקה נכין טבלה שתכלול בצד אחד את הפעולה המבוצעת ובצדה השני את מצב המונה והסכום.

הפעולה	מונה	סכום
1. אפס סכום		0
2. אפס מונה	0	
3. הוסף 1 למונה	1	
4. חבר מונה לסכום		1
5. האם מונה = 5 ? - לא		
6. הוסף 1 למונה	2 (=1+1)	
7. חבר מונה לסכום		3 (=2+1)
8. האם מונה = 5 ? - לא		
9. הוסף 1 למונה	3 (=1+2)	
10. חבר מונה לסכום		6 (=3+3)
וכו'		
- חבר 1 למונה	5 (=1+4)	
- חבר מונה לסכום		15 (=5+10)
- האם מונה = 5 ? - כן		
- <u>סוף</u>, התוצאה הנכונה <u>15</u>		

בעיה ד' - תרשים עם פעולות קלט/פלט

רוב העיבודים במחשבים אינם כה פשוטים. הדוגמא הבאה תציג טיפול ברשומות קלט פלט.

1. תאור הבעיה

א. נתון: אל הספריה של אוניברסיטה מסויימת מגיעים כל הספרים החדשים, כאשר אל כל אחד מהם מצורף כרטיס מנוקב. באוניברסיטה שלש פקולטות: מדעי הטבע, מדעי החברה ומדעי הרוח.

מבנה הכרטיסים:

תאריך	מחיר	שם המו"ל	שנת הוצאה לאור	שם המחבר	שם הספר	קוד המזמין

ב. העיבוד: יש לקרוא את הכרטיסים, להדפיס דו"ח שבועי לפי מבנה, הנתון להלן. יש להעלות את רשומת הקלט על סרט מגנטי לצורך העיבוד החדשי (חיוב מחלקות, זכוי בהנהח"ש וכד')

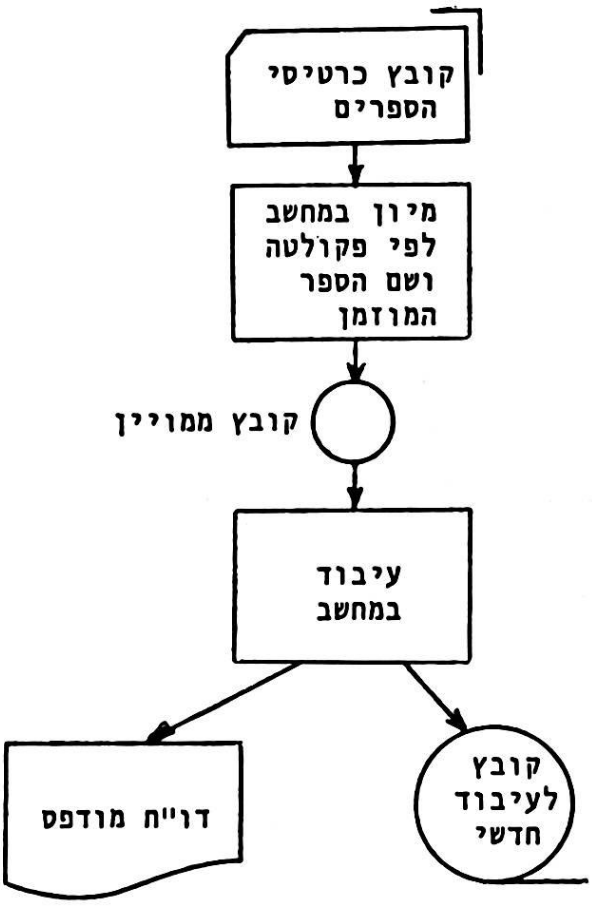
בתהליך העיבוד המוצג כאן אין מיון קודם של הקלט, לפיכך יודפס הפלט לפי סדר רשומות הקלט. הסיכומים ינתנו בסוף הדו"ח במרוכז.

ג. <u>פלט:</u> מבנה הדו"ח המודפס ניתן להלן:

ספריות באוניברסיטה - דו"ח שבועי			
הפקולטה המזמינה	שם המחבר	שם הספר	תאריך הזמנה

בסוף הדו"ח יש להדפיס את סה"כ הספרים, שנתקבלו עבור כל פקולטה באוניברסיטה (טבע, חברה או מדעי הרוח). אם חסר קוד הפקולטה בכרטיס - יש להדפיס כוכביות בשדה הפקולטה ולהדפיס "שגוי - חסר שם הפקולטה".

2. <u>תרשים הזרימה:</u>



3. <u>התרשים המלבני</u> יציג את שלבי העיבוד במחשב, הפקת הדו"ח והסרט לעיבוד החודשי. יש להדגיש את הנקודות הבאות:

1. <u>מונים:</u> בתחילת העיבוד אפשר להגדיר שדות, שימלאו תפקיד של מונה.
1 לפי נתוני הבעיה אפשר גם לדאוג לערך התחלתי, שמונים אלה יכילו. בדוגמה שלפנינו המונים יאופסו בתחילת העיבוד. יש לתת לכל מונה שם מיוחד, שונה מכל שם אחר בתכנית.

2. <u>עריכה:</u> כאשר הפלט הוא דו"ח מודפס. יש לערוך את השדות השונים כך, שהשורה תהיה נוחה לקריאה ותכיל בדיוק את הנתונים שרוצים להדפיס (לא פחות, אך גם לא יותר). העברת שדות שונים לשדות הפלט וסדורם עם מרווחים מתאימים - נקרא עריכה.

3. בפתרון שלפנינו התעלמנו מבעיית הכותרת ומספור הדפים.

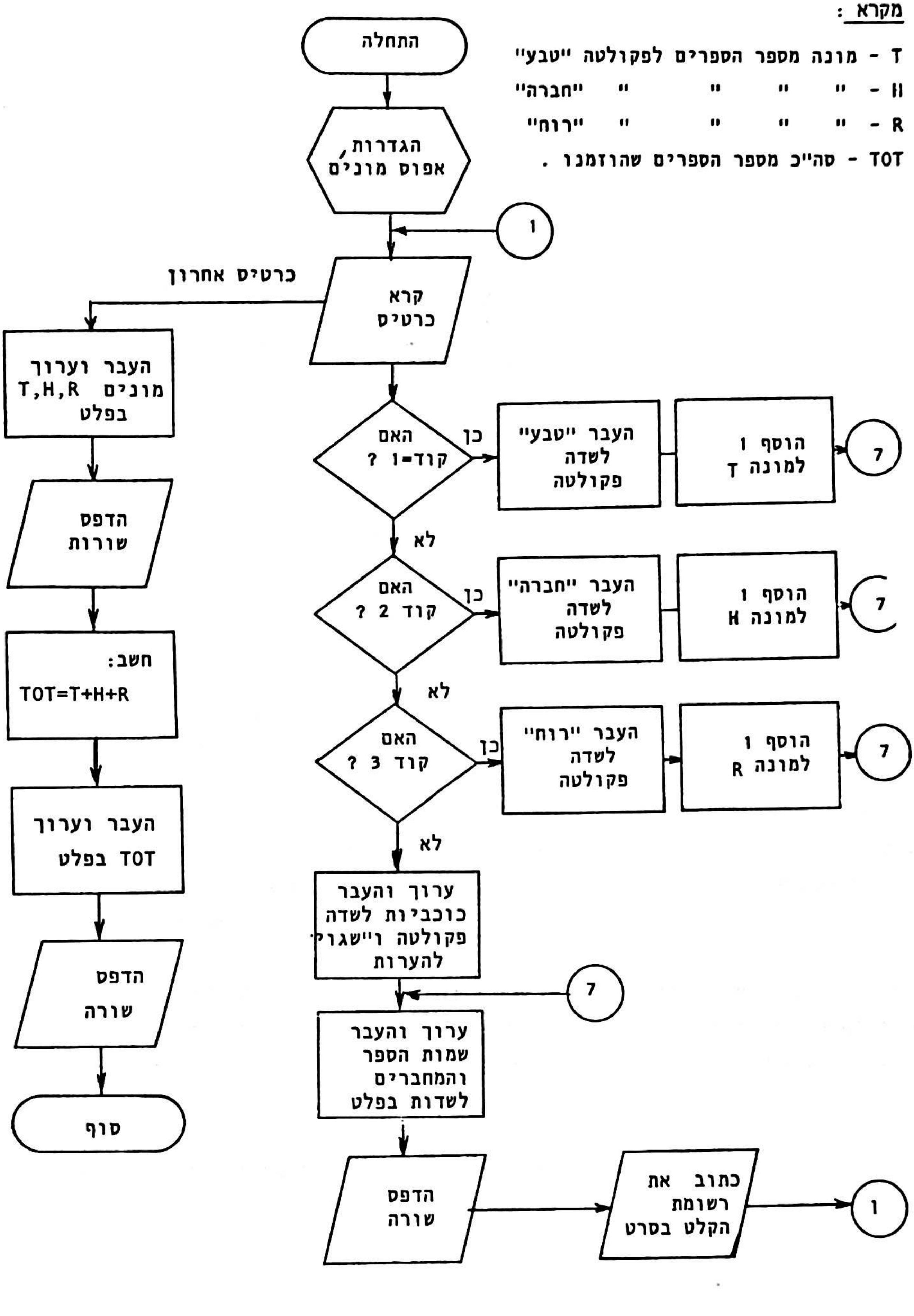
מקרא :

T - מונה מספר הספרים לפקולטה "טבע"

H - " " " "חברה"

R - " " " "רוח"

TOT - סה"כ מספר הספרים שהוזמנו .



4. **בדיקת התרשים המלבני**

כדי לבדוק אם התרשים, או הפתרון המוצג ע"י התרשים, עונה על תנאי הבעיה יש לבצע בדיקתו בעזרת נתוני נסוי.

בבעיה המוצגת ישנן פעולות עיבוד המבוצעות עם נתוני קלט. נתוני הפלט כוללים דו"ח מודפס שגם אותו יש לבדוק לפרטיו.

דוגמא לנתוני קלט לבדיקה:

קוד המזמין	שם הספר	שם המחבר	שנת הוצאה לאור	שם המו"ל	תאריך הזמנה	מחיר
1	אב 1	מחבר 1	1971	מול 1	1.8.74	15.-
3	אב 2	מחבר 2	1973	מול 2	5.8.74	30.-
2	אב 3	מחבר 2	1970	מול 2	6.7.74	12.-
1	אב 4	מחבר 5	1960	מול 4	8.8.74	14.-
4	אב 5	מחבר 6	1973	מול 1	9.8.74	15.-
3	אב 6	מחבר 3	1974	מול 3	9.8.74	10.-

5. דוגמא לדו"ח הפלט הצפוי בתהליך העיבוד

הפקולטה המזמינה	שם המחבר	שם הספר	מחיר
טבע	מחבר 1	אב 1	15.-
רוח	מחבר 2	אב 2	30.-
חברה	מחבר 2	אב 3	12.-
טבע	מחבר 5	אב 4	14.-
******	מחבר 6	אב 5	15.-
רוח	מחבר 3	אב 6	10.-

חיוב לפקולטה - טבע סה"כ 29.- ל"י

חיוב לפקולטה - חברה סה"כ 12.- ל"י

חיוב לפקולטה - רוח סה"כ 40.- ל"י

סה"כ 81.- ל"י

1 שגיאה במספר הפקולטה

בעיה הי - תרשים מלבני עם תהליכי חישוב רבים

הדוגמא הקודמת היתה אופיינית ל"עיבוד נתונים מינהלי". מכנים כך נוהל עיבוד בו פעולות קלט ופלט רבות ואילו החישובים ופעולות אחרות של יחידת העיבוד במחשב מועטות יחסית.

בדוגמא הבאה מוצגת בעיה מסוג שונה. בבעיה זו יש פעולות עיבוד רבות יחסית לפעולות הקלט והפלט.

1. תאור הבעיה

א. נתון: כרטסת עובדים במפעל. הכרטסת מכילה פרטים שונים על משכורות העובדים לצורך חישובי משכורת. הכרטסת מכילה כרטיס לכל עובד.

ב. עיבוד: יש לקרוא את קובץ הכרטיסים ולחשב:

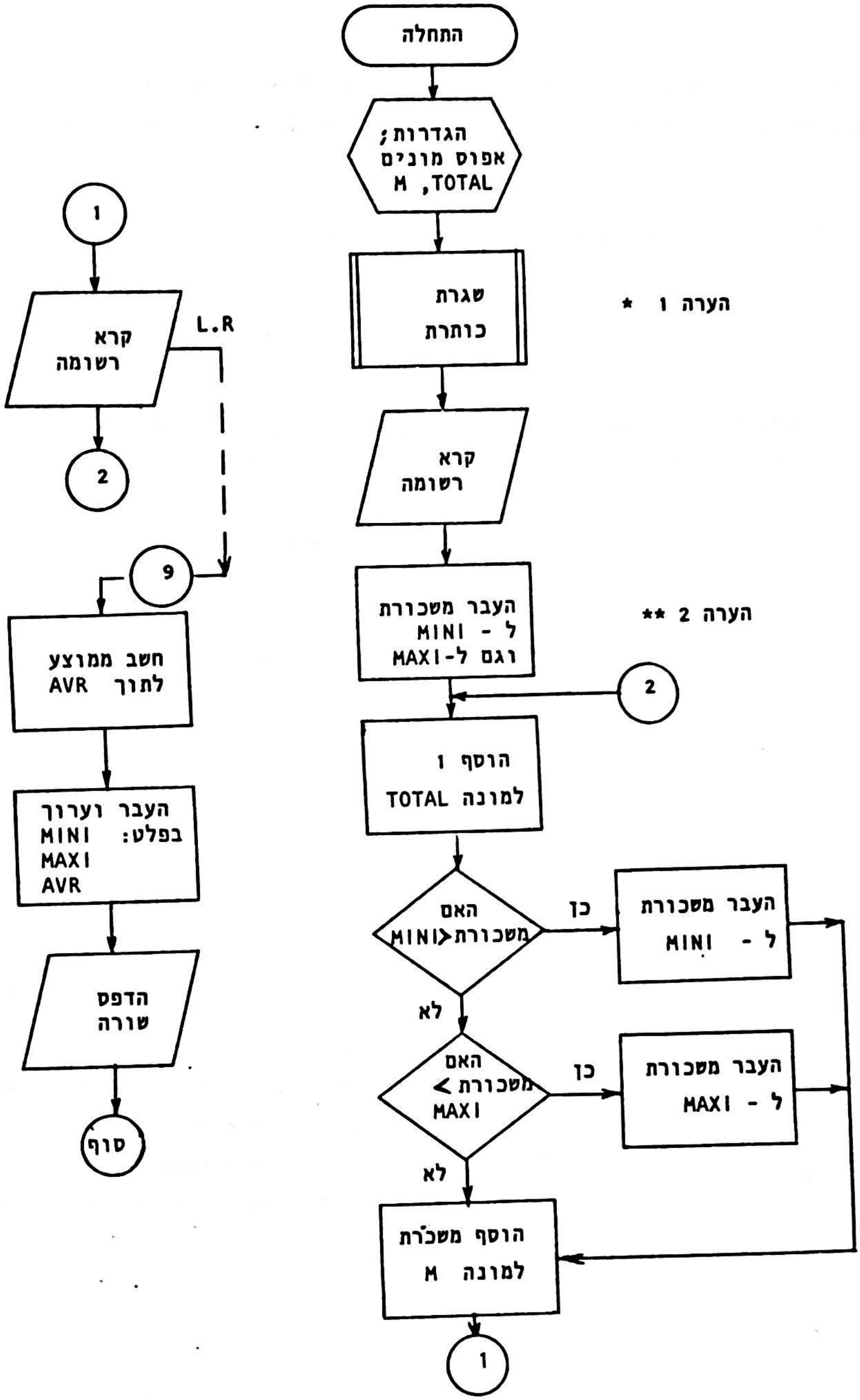
א) המשכורת הממוצעת במפעל.
ב) המשכורת הנמוכה ביותר.
ג) המשכורת הגבוהה ביותר.

ג. פלט: יש להדפיס את שלשת המספרים המבוקשים, עם כותרת מתאימה בראש הדף. הכותרת תכיל את תאריך העיבוד ושם הדו"ח.

2 התרשים המלבני

מקרא :

TØTAL - סיכום מספר העובדים במפעל
MINI - שדה לשמירת ערך המשכורת הנמוכה ביותר.
MAXI - שדה לשמירת ערך המשכורת הגבוהה ביותר
M - סיכום ערכי המשכורות של העובדים המפעל.
AVR - ערך המשכורת הממוצעת במפעל.


התחלה
הגדרות;
אפוס מונים
M ,TOTAL
שגרת
כותרת
הערה 1 *
קרא
רשומה
העבר משכורת
ל - MINI
וגם ל-MAXI
הערה 2 **
2
הוסף 1
למונה TOTAL
האם
משכורת<MINI
כן
העבר משכורת
ל - MINI
לא
האם
משכורת >
MAXI
כן
העבר משכורת
ל - MAXI
לא
הוסף משכורת
למונה M
1
1
קרא
רשומה
L.R
2
9
חשב ממוצע
לתוך AVR
העבר וערוך
בפלט: MINI
MAXI
AVR
הדפס
שורה
סוף

3. הסברים

הערה 1: שגרה (רוטינה)

בתחילת התרשים מופיעה צורה הנדסית חדשה: מלבן עם שני קווים בצד. זהו סימן לשגרה (רוטינה) שהיא סדרת הוראות, שפירוטן מופיע במקום אחר. להכנסת השגרה לתרשים מטרה כפולה: מצד אחד היא מפשטת את התרשים במקום לסרבל אותו, ומצד אחר השגרה היא סדרת פעולות שעשויות להתבצע מספר פעמים בתכנית וע"כ אפשר לחסוך את כתיבתן החוזרת. בדוגמה דלעיל נתיחס למקרה הראשון (מניעת סירבול) פרוט השגרה:

1) נקוי שטח ההדפסה.
2) הוספת 1 למונה ועריכתו בשטח ההדפסה.
3) העברת ועריכת הכותרת בשטח ההדפסה (כולל תאריך)
4) הדפסת הכותרת.
5) נקוי שטח ההדפסה.

הצגת השגרה כקטע תכנית, שיתבצע מספר פעמים תובא מאוחר יותר.

אם הדו"ח גולש מעל קיבולת של דף אחד תהיה פנייה לשגרת הכותרת בכל מעבר אל דף חדש.

הערה 2:

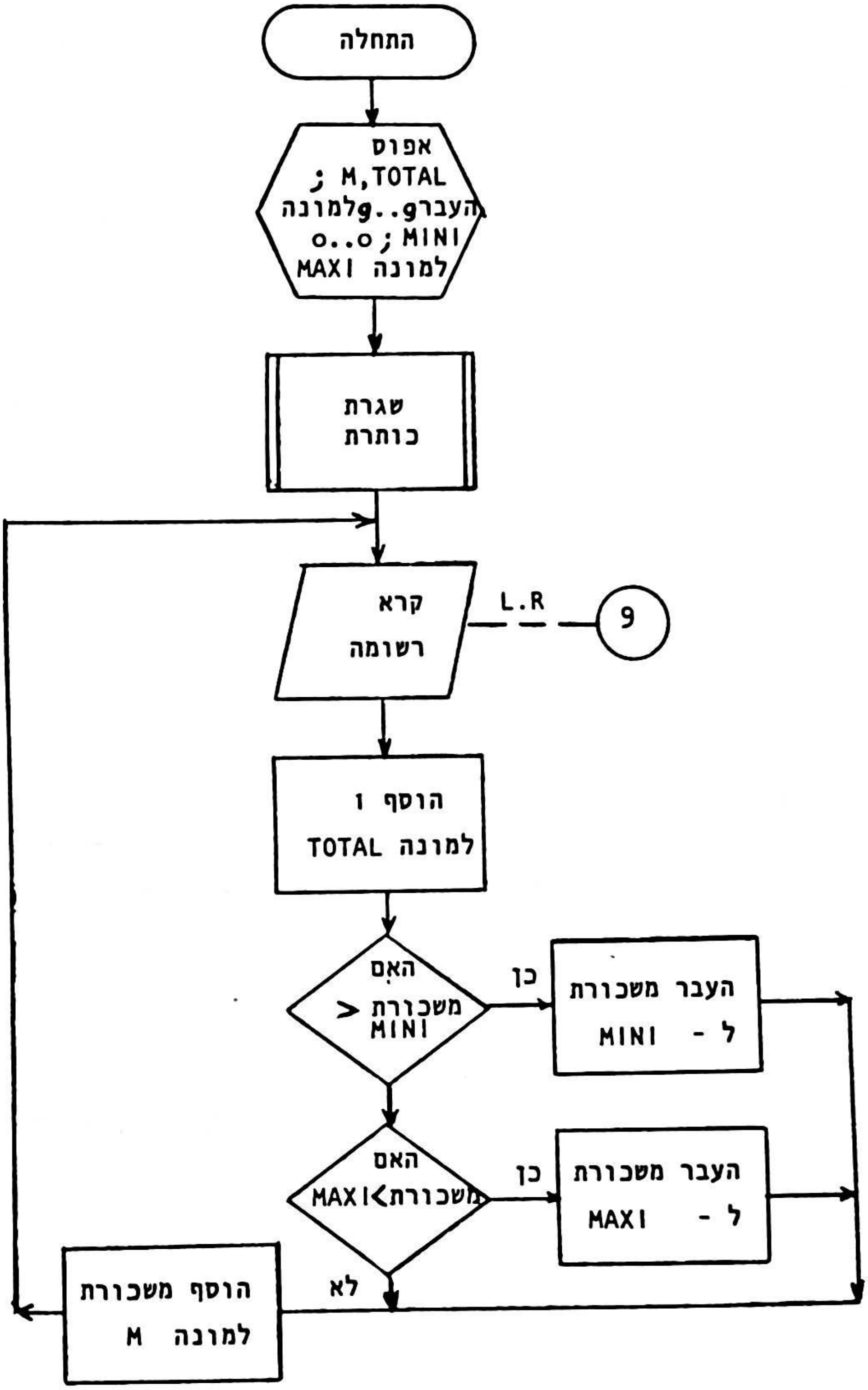
העברת המשכורת ל - MINI ול-MAXI . במשך העיבוד המשכורת הנמוכה ביותר נמצאת בשדה MINI והמשכורת של כל עובד חדש מושוית אל תוכן שדה MINI, אם המשכורת הנבדקת נמוכה מתוכן השדה MINI היא מועברת ל - MINI . (התהליך דומה לגבי מציאת המשכורת הגבוהה ביותר ולהעברתו ל - MAXI).

אולם קיימת הבעיה של המשכורת הראשונה , אותה לא נוכל להשוות למשכורות קודמות. במקרה כזה נעביר את המשכורת הראשונה ל - MINI, כי עד המשכורת הראשונה ועד בכלל היא הנמוכה ביותר, ומקומה לכן ב - MINI .

לפי אותם שקולים המשכורת הראשונה היא גם המשכורת הגבוהה ביותר, לכן נעביר אותה גם ל - MAXI .

ג. כדי להמנע מבצוע פעולות קריאה אחדות באותה תכנית אפשר להכין מראש את ערכי המונים MAXI ו - MINI כדי שאפשר יהיה להשוות אליהם את הרשומה הראשונה.

התרשים המלבני של הבעיה יהיה כעת:



<u>בעיה ו' - זהוי קבוצה</u>

קיימים יישומים רבים, בהם העיבוד מתבטא בקריאת <u>רשומות רבות בעלות אותו מספר זהוי ובהדפסת שורת סכום אחת</u>, לכל קבוצה וקבוצה.

1. <u>תאור הבעיה</u>

א. המכון לרווחה חברתית עורך מחקר על ההוצאות וההכנסות של משפחות משכבות חברתיות שונות באוכלוסיה.

בשלב מסויים של המחקר רוצים לחשב את ההוצאות הממוצעות לנפש במשפחה, בתקופה מסויימת. ראש המשפחה נתבקש לרשום כל הוצאה במסמך נפרד ולהגיש למכון.

ב. במכון ניקבו את המסמכים בכרטיסים במבנה הבא:

ההוצאות בל"י	מספר הנפשות במשפחה	כתובת	שם	מספר ת.ז.

לאחר המיון עוברים הכרטיסים, אל המחשב לשם עיבוד. בסוף העיבוד יש להדפיס <u>שורה אחת</u> לכל משפחה לפי המבנה הבא:

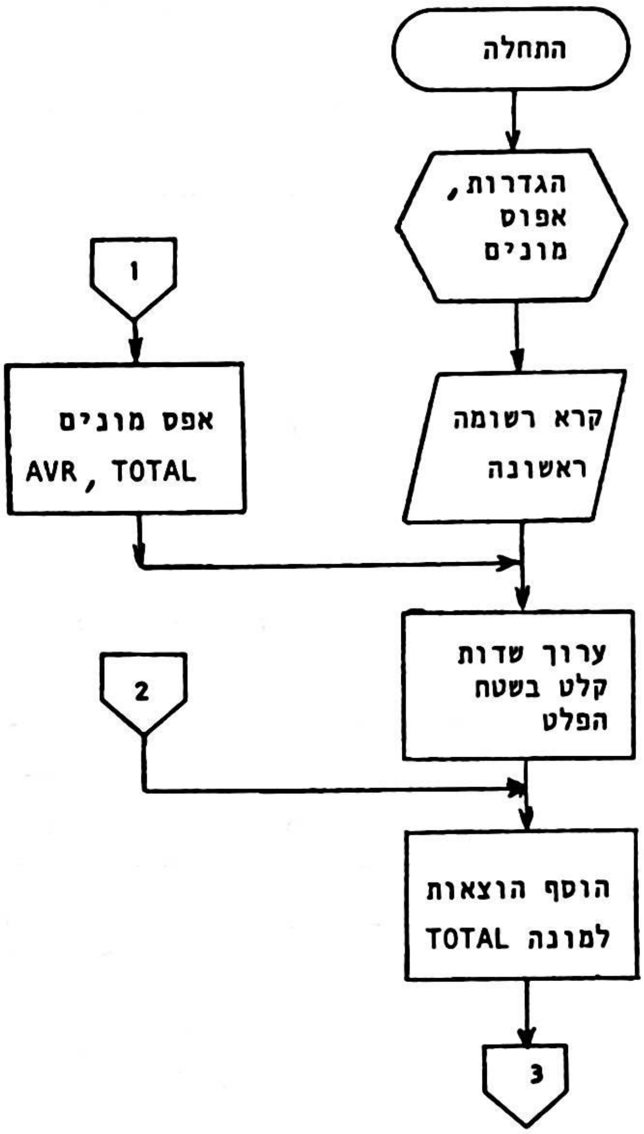
ד ו " ח ה ה ו צ א ו ת				
מס' זהוי	ש ם	מס' נפשות	ס"ה הוצאות	הוצאה ממוצעת

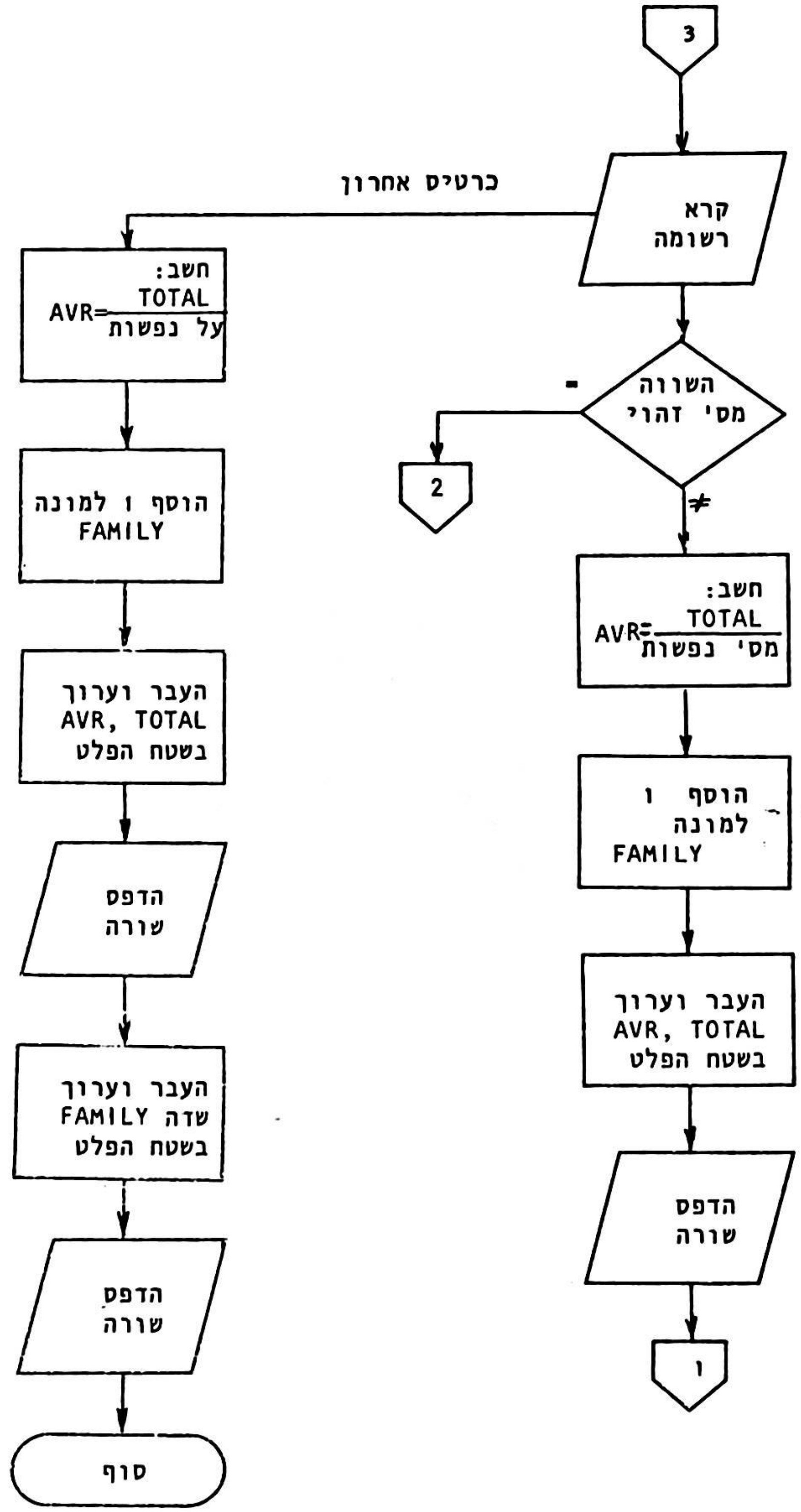
בסוף הדו"ח יש להדפיס שורה, ובה מספר המשפחות שהשתתפו בשלב זה של המחקר.

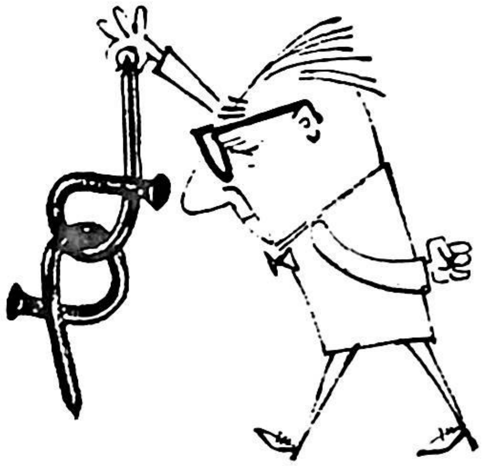
2. <u>תהליך הפתרון</u>

במקרה שלפנינו הקלט מורכב מקבוצות - קבוצות של כרטיסים, כאשר כל קבוצה מזוהה עם מספר הזהוי של ראש המשפחה. בזמן קריאת הכרטיסים נעביר את תוכן הכרטיס הראשון של הקבוצה לפלט, וכל הכרטיסים שיבואו אחריו - יושוו (ע"י מספר הזהוי - מכיוון שהם ממויינים) אל המספר של הכרטיס הראשון. אם מתברר שהכרטיסים שייכים לקבוצה אחת, נצבור את הסכומים במונים. אם יתגלה כרטיס עם מספר זהוי שונה (המסמל קבוצה חדשה - משפחה חדשה) אזי נפסיק לצבור את ההוצאות, נחשב ונדפיס את המידע הדרוש לגבי הקבוצה שזה עתה נסתיימה. רק לאחר מכן נפנה לכרטיס החדש, המסמל כאמור קבוצה חדשה.

3. <u>תרשים מלבני</u>




3
כרטיס אחרון
קרא
רשומה
השווה
מסי זהוי
=
2
≠
חשב:
AVR= TOTAL / מסי נפשות
הוסף 1
למונה
FAMILY
העבר וערוך
AVR, TOTAL
בשטח הפלט
הדפס
שורה
1
חשב:
AVR= TOTAL / על נפשות
הוסף 1 למונה
FAMILY
העבר וערוך
AVR, TOTAL
בשטח הפלט
הדפס
שורה
העבר וערוך
שדה FAMILY
בשטח הפלט
הדפס
שורה
סוף



לכל בעיה — פתרון !

חלק שלישי:

תכנות פורטרן

FORTRAN

פרק ו: מרכיבי שפת פורטרן

בפרק זה נדון בשפת התכנות הקרויה פורטרן (FORTRAN). שם זה נגזר מראשי התיבות של FORmula TRANslator - "מתרגם נוסחאות".

שם שפת התכנות מצביע על כך שמדובר בשפה לתכנות מדעי. ואכן, כללי הכתיב של השפה מזכירים את הכתיב המתמטי המקובל.

פרק זה מכוון לערוך "פגישת היכרות" עם השפה ולתת לקורא את ההבנה והיכולת לדעת "מה" ו"איך" אפשר לעשות בעזרת שפה זו. הקורא המעונין להרחיב והלעמיק ידיעותיו בתכנות פורטרן ימצא ספרות ענפה בנושא הן בעברית והן באנגלית.

ניבים ואלפבית של השפה

שפת פורטרן הינה שפת תכנות עילית (High level programming language). להאורה, שפת פורטרן אינה תלויה במחשב מסויים, ותוכנית אשר נכתבה בפורטרן ניתנת להרצה בכל מחשב שיש לו מהדיר (compiler) של השפה הזאת. אולם למעשה, אין הדבר אפשרי מכיון שיש פקודות מסויימות שהן תוספות שהוסיפו חברות מחשבים שונות לשפה; לכן אפשר לומר שיש לפורטרן מספר "ניבים". על כן יש צורך לפנות לחוברת הפורטרן המתאימה למחשב שבו מתכוונים לעבוד.

הניב אשר נתרכז בו קרוי פורטרן IV (FORTRAN IV) והוא המקובל כשפת הפורטרן התיקנית.

כמו לכל שפה אחרת כן גם לפורטרן יש אלף-בית ממנו מרכיבים את המילים, המספרים, הביטויים והמשפטים הכלולים בשפה. האלף-בית של פורטרן כולל את הסימנים הבאים:

סימנים אלפבתיים: A B C D E F G H I J K L M N Ø P Q R S T U V W X Y Z

שים לב, כותבים רק באותיות גדולות. על האות O (או) רושמים לוכסן כדי להבדילה מהספרה 0 (אפס).

סימנים מספריים: 0 1 2 3 4 5 6 7 8 9

סימנים מיוחדים: $ ( ) , . / * - +

בניב פורטרן של י.ב.מ. נכלל גם הסימן ' (גרש).

עתה ניגש להכיר כמה מהמרכיבים החשובים ביותר של שפת פורטרן אשר יאפשרו להבין ולכתוב תוכניות פשוטות בשפה זו.

מספרים קבועים ומשתנים

שיטות להצגת מספרים

בפרק זה נדון בשתי שיטות חישוב: שיטת הנקודה הקבועה (Fixed Point) ושיטת הנקודה הצפה (Floating Point).

בשיטת הנקודה הקבועה מאוחסנים המספרים במלת המחשב כמספרים שלימים. מיקומה של הנקודה העשרונית הוא מימין למספר. הצגה זו מגבילה את תחום המספרים, בהתאם ל"גודלה" של מילת-המחשב.
ע"י שינוי יחידת המדה אפשר להציג כל מספר כמספר שלם, כמו למשל להציג משקל בגרמים במקום בקילוגרמים. החישובים מתבצעים בדרך דומה לדרך הבצוע של חשובים אריתמטיים.
כדי להתגבר על בעיה זו נוצרה שיטת הנקודה הצפה. בשיטה זאת מיצגים את המספר על-ידי מנטיסה ומעריך (ראה הסבר מושגים אלה להלן). מילת המחשב מחולקת לשניים: חלק אחד מכיל את המעריך והחלק השני מכיל את המנטיסה. שיטה זו מאפשרת לאחסן תחום גדול יותר של מספרים אם כי מפסידים דיוק, במספר הספרות המשמעותיות. השיטה דורשת מעגלים אלקטרונים שונים מאלה אשר דרושים בחישוב בשיטת הנקודה הקבועה..

בפורטרן וגם במחשב מיוצגים כל המספרים בעלי נקודה עשרונית בשיטה הנקודה הצפה.

ההפרדה הבסיסית היא בין המספרים השלימים (Integers) שהחישובים בהם נעשים בשיטת הנקודה הקבועה לבין המספרים הממשיים (Real numbers) אשר החישובים בהם נעשים בשיטת הנקודה הצפה.

ישנה הבחנה נוספת בהצגת המספרים: בין קבועים שהם ערכים מספריים אשר לא משתנים בזמן ביצוע החישובים, לבין משתנים אשר ערכם יכול להשתנות בעת ביצוע

התוכנית. ההבחנה האחרונה זהה למעשה לזאת המקובלת באלגברה.

בהצגת המספרים נבדיל, אם כן, בארבעה הסוגים הבאים: קבועים שלמים, קבועים ממשיים, משתנים שלמים, ומשתנים ממשיים.

א. קבועים שלמים (Integer Constants)

קבועים אלו מיוצגים על-ידי סדרה של ספרות ללא נקודה עשרונית. המספר המקסימלי המותר של ספרות שונה במחשבים השונים, והוא נע בין 10 ל-14.
דוגמאות לקבועים שלמים: 2315 , 7987 , 132. סימן ה"-" (מינוס) חייב לבוא לפני המספר כאשר המספר הוא שלילי. למשל: 735- , 15- .
אם המספר הוא חיובי יכול לבוא לפניו סימן ו "+" אך אין הכרח לרשום אותו. כך למשל 127+ ו 127 מייצגים אותו מספר, וייקלטו ע"י המחשב בצורה זהה.

המושג "פסיק" (,) איננו קיים בהצגת מספרים כדי להדגיש את האלפים וכד'. יש לשים לב לעֻובדה שנקודה עשרונית אסור שתופיע בהצגה של קבוע שלם. חסרונה של הנקודה העשרונית הוא אשר מורה למחשב שהנתון הוא קבוע שלם ושהחישובים ייעשו בשיטת הנקודה הקבועה. זכור: בפורטרן יכולה להיות למספרים 3.0 ו-3 משמעות שונה אם כי ערכם שוה.

ב. קבועים ממשיים (Real Constants)

הקבועים הממשיים מיוצגים ע"י סדרה של ספרות עם נקודה עשרונית. גם כאן המספר המקסימלי של הספרות משתנה ממחשב אחד למשנהו. הכללים לגבי סימן ה-"+" וסימן ה-"-" הם כמו לגבי קבועים שלמים.
דוגמאות לקבועים ממשיים: 17.8+ , 1392.5- , 15.7 , 0.723.
הנקודה העשרונית יכולה לבוא לפני המספר: למשל 513. אשר פירושו הוא 0.513, או אחריו: 1725. והכוונה ל-1725.0. קיומה של הנקודה העשרונית מורה שהמספר "ממשי" והחישובים אשר מספר זה קשור בהם יבוצעו בשיטת הנקודה הצפה.

דרך נוספת להצגת המספרים הממשיים היא ההצגה המעריכית (אקספוננציאלית). את המספר 173000000000 נוכל להציג באריתמטיקה המקובלת כ-1.73×10^{11}. המספר הכופל את החזקה של 10 קרוי מנטיסה, והחזקה אשר בה יש להעלות את 10 קרויה בשם מעריך (אקספוננט). הצגה דומה להצגה האלגברית המתוארת לעיל קיימת גם בפורטרן:

המספר 1.73×10^{11} ירשם בפורטרן כך: 1.73E11. E מסמלת את בסיס החזקה 10, המנטיסה נרשמת לשמאלה של האות E והמעריך נרשם לימינה. המעריך הוא מספר שלם ואין להוסיף לו נקודה עשרונית. אם המעריך שלילי, יבוא לפניו הסימן "-" (מינוס). למשל המספר 0.97×10^{-12} ירשם כ- 0.97E-12. מעריך חיובי אפשר, אך אין הכרח, לציין ע"י הסימן "+". כך למשל 721.2E+07 ו-721.2E07 ו-721.2E7 מייצגים אותו המספר.

בשיטת ההצגה המעריכית במחשב מקבלת המנטיסה ערך קטן מ-1, ובהתאם לכך נקבע ערכו של המעריך. המשתמש מציג את המעריך לפי חזקות של 10 אולם במחשב מוצג המעריך לפי חזקות של 8,2 או 16.

להלן מספר דוגמאות נוספות של מספרים ממשיים הכתובים בצורה מעריכית:

ערכו של המספר	הצגה בצורה מעריכית	הצגה במחשב
0.001	1.0E-3	0.1E-2
72.389	7.2389E1	0.72389E2
123.7	123.7E0	0.1237E3
-7.95×10^{18}	-7.95E18	-0.795E19
1000	1.0E3	0.1E4

.. משתנים שלימים (Integer Variables)

המשתנים השלימים מיוצגים על-ידי צירוף של אחת עד שש אותיות ו/או ספרות, ואסור להכליל בשם של משתנה סימנים מיוחדים. התו הראשון המרכיב את שמו של המשתנה השלם חייב להיות אחת מהאותיות N, M, L, K, J, I. שאר התוים יכולים להיות צירוף כלשהו של אותיות וספרות.

דוגמא לשמות חוקיים של משתנים שלימים:

ICØUNT; K ; J12 ; N2RF

הערה: בניבים אחדים של פורטרן (למשל ב-CDC) יכול המשתנה להיות בן 7 תוים.

ד. משתנים ממשיים (Real Variables)

הכלל לבניית שמות של משתנים ממשיים דומה לזה של המשתנים השלימים. ההבדל היחיד הוא שהתו הראשון בשם של משתנה ממשי חייב להיות שונה מהאותיות

N, M, L, K, J, I.

למשל: ALPHA ; RO ; COEF ; X ; AB57N.

כל אלה הם שמות חוקיים של משתנים ממשיים.

פעולות חשבון

פעולות החשבון "המוכרות" לפורטרן הן הפעולות האלגבריות הרגילות: חיבור, חיסור, כפל, חילוק והעלאה בחזקה. אולם לא כל סימני הפעולות זהים לאלה של האלגברה. סימני הפעולות בפורטרן הם:

+ חיבור
- חיסור
* כפל
/ חילוק
** העלאה בחזקה

סכום של שני מספרים A ו-B מיוצג בפורטרן ע"י A+B, הפרש ע"י A-B, ומכפלתם תכתב כך A*B. חובה לרשום את סימן הכפל שלא כמו באלגברה, בה מותר לרשום AB . המנה A:B תכתב כ-A/B ואילו A^B (A בחזקת B) ירשם כך A**B.

לא קיימים סימנים מיוחדים להוצאת שורש למשל. פעולה זו תוצג ע"י פעולת העלאה בחזקה כמו A**0.5 כדי לבטא שורש רבועי של A. פעולות כמו אינטגרל או סיכום סדרת מספרים יוצגו ע"י פונקציה מתאימה, כפי שיוסבר בהמשך.

ביטויים אריתמטיים

כללי הבנייה של ביטויים אריתמטיים בפורטרן דומים מאד לכללי הבנייה של ביטויים אריתמטיים במתימטיקה. נתבונן בהשוואה בין מספר ביטויים הכתובים בכתיב המתימטי המקובל ובכתיב של פורטרן:

	פורטרן	כתיב מתימטי מקובל
1	A+B-3.0*C	$A+B-3C$
2	A**2+B**2	A^2+B^2
3	(A*B)**0.5	$\sqrt{A\cdot B}$
4	0.5*SUM	$\frac{1}{2}SUM$
5	I+J**2+3*K	$I+J^2+3\cdot K$
6	(A+2.)/CØUNT	$\frac{A+2}{CØUNT}$

אפשר להשאיר תו ריק (blank) בין משתנה לבין סימן הפעולה החשבונית או סוגרים, כדי להקל על קריאת הביטוי. אסור כמובן לעשות זאת בין אותיות של שם משתנה או בין שתי הכוכביות של סימן העלאה בחזקה.

דומה שטבלה משווה זו מוכיחה שאכן הצגת ביטויים בפורטרן היא "טבעית" למדי. עם זאת, יש ללמוד את הכללים השונים הקשורים בבניה נכונה של ביטויים אריתמטיים - וכל זאת כדי שנוכל להגדיר למהדיר במדויק מה כוונתנו, ולאיזה ביטוי אנו מתכוונים.

<u>כלל 1:</u> כל הקבועים והמשתנים בביטוי אריתמטי חייבים להיות מאותו סוג: כולם שלימים או כולם ממשיים. היוצא מן הכלל היחיד יצויין בכלל מספר 2. בניבים רבים של פורטרן, כמו למשל אלו הפועלים במערכות 360 ו-370 של IBM מרשים מידה מסויימת של ערבוב שלימים וממשיים באותו ביטוי.

הבה נחזור לדוגמאות שראינו קודם. בשורה הראשונה מופיע הביטוי 3·C תיבתו של ביטוי זה בפורטרן תהיה 3.0*C (או 3.*C) וזאת משום העובדה ש-C הוא משתנה ממשי. לכן, לפי כלל מספר 1 יש לרשום את הקבוע 3 בצורה של קבוע ממשי, כלומר 3.0 או 3. .

בדוגמא מספר 5 מופיע 3·K אשר יכתב בפורטרן כ-3*K. מכיון ש-K הוא משתנה שלם ירשם גם הקבוע 3 כקבוע שלם, כלומר: ללא הנקודה העשרונית.

<u>כלל 2:</u> בפעולת העלאה בחזקה, יכול מעריך החזקה להיות מספר שלם או מספר ממשי מבלי להתחשב במספר שמעלים בחזקה. זאת שלא כמו בכלל 1, המחייב הצגה אחידה של המספר.

לדוגמא: כל הבטויים האלה הם חוקיים

A**5   CØUNT**R   N**0.5   I**J   . B**K

<u>כלל 3:</u> הפעולות בפורטרן מתבצעות לפי סדר העדיפויות הבא:

א. העלאה בחזקה

ב. כפל וחילוק

ג. חיבור וחיסור

פירוש הדבר שאם לא ניתנת למהדיר כל אינפורמציה נוספת על הסדר בו עליו לבצע את הפעולות (בעזרת סוגריים, למשל, כפי שמוסבר בכלל הבא) יבוצעו פעולות החזקה, אחריהן פעולות הכפל והחילוק ולבסוף פעולות החיבור והחיסור. פעולות

בעלות אותה העדיפות מתבצעות משמאל לימין.

הבה נדגים כלל זה. יהיו A=5.0, B=2.0, C=7.0. הבטוי B-3.*C יחושב בשלבים הבאים:

(1) 3.*7 = 21.0 (עדיפות הכפל גבוהה מזו של החיבור והחיסור)

(2) 2.0-21.0 =-19.0

הביטוי A/B*C אשר בו הפעולות הן באותה עדיפות יחושב משמאל לימין. כלומר:

(1) 5./2. = 2.5

(2) 2.5*7. = 17.5

נתבונן שוב בדוגמא זו. הפעולות שהן בעלות אותה עדיפות חושבו משמאל לימין. הביטוי A/B*C מתאים אם-כן לביטוי $\frac{A}{B}C$ בכתיב המתימטי המקובל. אבל אילו רצינו לחשב את $\frac{A}{B\cdot C}$ הרי שהיינו צריכים להשתמש בסוגריים כדי לשנות את סדר העדיפויות ולרשום את הביטוי כך: A/(B*C). השימוש בסוגריים מוסבר בכלל הבא.

כלל 4: משתמשים בסוגריים כדי לציין פעולות שיש לבצען תחילה כפי שמקובל באריתמטיקה. סדר הפעולות הנתון ע"י כלל 3 ניתן אם-כן לשינוי בהתאם לרצונו של המתכנת. אם יש "סוגריים בתוך סוגריים", יחושב תחילה הביטוי בסוגריים הפנימיים ואחריו הביטוי אשר בסוגריים החיצוניים. לאחר שחושבו הביטויים בסוגריים ימשיך המחשב לבצע את הפעולות החשבוניות לפי כלל 3.

דוגמא: כדי לחשב את הביטוי 2.*((A+B)**2+C**3) - 7.*(A-D)/3.

כאשר: A=1., B=2., C=3., D=4. יבצע המחשב את הפעולות הבאות:

(1)	1.+2.=3.	(A+B)
(2)	3.**2=9.	(A+B)**2
(3)	3.**3=27.	C**3
(4)	9.+27.=36.	(A+B)**2+C**3
(5)	2.*36.=72.	2*((A+B)**2+C**3)
(6)	1.-4.=-3.	A-D
(7)	7·(-3.)=-21.	7*(A-D)
(8)	-21./3=-7.	7*(A-D)/3.
(9)	72.-(-7.)=79.	2*((A+B)**2+C**3)-7*(A-D)/3.

כלל 5: סימן של פעולה חשבונית איננו יכול להקדים את סימן ה-"+" או סימן ה-"-". אבל מותר להשתמש בסוגריים כדי להקיף את הקבוע או המשתנה אליו מתייחס הסימן.

לפי כלל זה אסור:

A*-B , A+-B, I-+J, A/-B

ויש להחליפם ב:

A*(-B), A+(-B), I-(+J), A/(-B)

משפטי השמה ומושג השויון

הביטוי האריתמטי הוא צרוף של איברים הקשורים ביניהם בפעולות חשבון. לאחר בצוע החישוב של הביטוי יש להקנות את הערך המתקבל למשתנה כלשהוא כדי שאפשר יהיה להמשיך ולהתיחס לערך זה בהמשך.

קביעת ערך של ביטוי למשתנה נעשית ע"י הוראה הנקראת "משפט השמה" (Assignment statement).

מבנה משפט ההשמה דומה למשפט השויון האלגברי וצורתו היא:

משתנה = בטוי אריתמטי

בצד השמאלי של השויון יופיע רק משתנה אחד. "הביטוי" יכול להיות גם הוא משתנה או מספר בודד.

לדוגמא: M = 5

A= CØUNT + SUM/2.5

פרושו של מושג השויון (=) במשפט ההשמה הוא: חשב את ערך הביטוי המופיע בצד ימין של ההוראה וקבע למשתנה אשר מופיע בצד שמאל את הערך שחושב.

למשל, אם למשתנה בשם I יש הערך 3 ובתוכנית יופיע המשפט K=I+7 תתבצע הפעולה הבאה: הביטוי בצד ימין יחושב כ-3+7, היינו 10. ותוצאה זו תהיה הערך החדש של המשתנה K.

באופן דומה: ערכו של משתנה בשם SUM הוא 5.0 וערכו של משתנה בשם X הוא 7.0 יש לחשב את

SUM = SUM + X

הפעולה הראשונה תהיה חישוב SUM+X, כלומר .5.+7.=12, ולאחר מכן - קביעת ערך זה למשתנה SUM. שים לב שבמשפט זה ניתן הערך החדש למשתנה SUM והוא שינה

את ערכו הראשון. אם בהמשך התוכנית תופיע הוראה זהה SUM=SUM+X, תהיה הפעם התוצאה חיבור של 12. ל-7. והתוצאה 19. תקבע כערכו החדש של SUM.

בשיטה זו משתמשים כדי ליצג מונה: אם חוזרים על ההוראה M=M+1 או I=I+1 פעמים אחדות, היא גורמת בכל חזרה לשנוי (או העלאת) ערכו של המשתנה ב-1.

בדרך כלל יהיו כל המספרים והמשתנים במשפט בעלי אותה שיטת יצוג - מספרים שלימים או ממשיים. החישוב של הביטוי יבוצע לפי שיטת הייצוג של המספרים המו-פיעים בו והערך הסופי ינתן למשתנה שהוא בעל שיטת יצוג זהה או שונה.

כאשר מספר שלם מועבר למשתנה ממשי, ערכו של המספר לא נפגע. אולם, כאשר מעבירים מספר ממשי למשתנה המייצג מספר שלם נקטע החלק העשרוני של המספר.

לדוגמא: - אם B=2.6 וכותבים N=B יהיה ערכו של N בסוף הפעולה - 2 בלבד.

- אם מחלקים מספרים שלימים מאבדים את החלק העשרוני של התוצאה גם אם מעבירים אותה למשתנה ממשי.
- אם מחלקים שני מספרים ממשיים ומעבירים את התוצאה למשתנה שלם - מאבדים את החלק העשרוני של התוצאה.

לדוגמא: A=5, B=3 ומחשבים M=A/B ערכו של M יהיה 1.

לאחר הכרת אבני היסוד של השפה: האלף-בית, המילים (קבועים ומשתנים), הפעלים (פעולות החשבון) והמשפטים (משפטי-השמה) אפשר לכתוב תוכנית לפתרון בעיה.

ת ר ג י ל:

יש לכתוב תוכנית לפתרון המשואה הרבועית

$$ax^2+bx+c=0$$

כאשר a=2., b=3., c=-1.

פתרון המשואה הוא

$$x_{1,2}=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

את הביטוי $\sqrt{b^2-4ac}$ מקובל לכנות "דלתא".

כדי לכתוב את התוכנית בפורטרן יש לכתוב את שמות המשתנים השונים באו-תיות גדולות. כמו כן רושמים שמות משתנים X1 ו-X2 בהתאמה עבור השורשים (אפשר לרשום גם שמות אחרים) כאשר הספרות הן בגובה האות ולא כמו בסימון אינדקס.

התרת הבעיה בפורטרן תיכתב בצורה הבאה:

```
A=2.
B=3.
C=-1.
DEL=(B**2-4.*A*C)**0.5
x1=(-B+DEL)/(2.*A)
x2=(-B-DEL)/(2.*A)
```

נסה נא להבין בעצמך את הכתוב ושים לב לדמיון שבין כתיב הפורטרן לכתיב האלגברי המקדבל.

הוראות התוכנית מציגות בשלבים את דרך הפתרון והן מבוצעות בזו אחר זו לפי סדר כתיבתן. ההוראות הרשומות כאן מכוונות לפתרון המשואה והן אינן מייצגות עדיין תוכנית שלימה. שלשת המשפטים הראשונים של התוכנית קובעים את הערכים של מקדמי המשואה הרבועית:

ההוראה .A=2 קובעת למשתנה בשם A את הערך .2.
וההוראות .B=3; .C=-1 קובעות את הערכים של המשתנים B ו-C בהתאמה.

המשפט הרביעי מחשב את הביטוי $\sqrt{b^2-4ac}$ וקובע את ערכו למשתנה בשם DEL.

ולבסוף, שתי השורות האחרונות מחשבות את שני השורשים של המשוואה ומאחסנות את התשובות במשתנים X1 ו-X2 בהתאמה.

משפטי בקרה

הביצוע הרגיל של הוראות התוכנית הוא לפי סדר כתיבתן, בזו אחר זו.
לעיתים יש צורך לחרוג מסדר ההוראות כדי לבצע הוראות אחרות, להפסיק את ביצוע התוכנית או לבצע פעולות אחרות

ביצוע פעולות אלו נעשה ע"י משפטי בקרה (control statements), אשר כו-ללים הוראות שונות: הוראות דילוג, משפטי תנאי, הוראות עצור והוראות המשך ועוד.
הפתרון שהוצג עד כה למשוואה הריבועית אינו שלם. נבחרו מקדמים A,B,C כאלה שעבורם $B^2 - 4AC > 0$, כלומר, למשוואה אכן יש שני פתרונות. לפני ביצוע החישוב

יש לבדוק את ערכו של ביטוי זה כדי לדעת אם ניתן לחשב ממנו את השורשים, אם לאו. אפשר להטיל על המחשב את ביצוע הבדיקה, ולפעול בהתאם לתוצאתה. כלומר לעבור לקטע תוכנית אשר יבצע את החישוב המתאים כאשר תנאי מסויים מתקיים.

לשם ביצוע הבדיקה: השואה, החלטה וסיעוף לקטעה תוכנית נדרש יש צורך להעזר במשפטי הבקרה המתאימים. כדי שאפשר יהיה "לעבור" או "להסתעף" לקטע תוכנית אם מתקיים תנאי מסויים יש צורך לזהות את ההוראה הראשונה בסדרת הוראות זו.

הזהוי נעשה ע"י מספר הוראה או תוית (label). מספר ההוראה, או התוית, הוא מספר שלם בן 5 ספרות לכל היותר, והוא ירשם לשמאלה של ההוראה עצמה.

בדרך כלל תנתן תוית רק להוראות שמעונינים בזיהויין, וכמובן שאין לתת לשתי הוראות את אותה התוית. לא חייב להיות כל סדר בכתיבת מספרי ההוראות. הוראה עם תוית יכולה להיראות כך:

```
5      A=B**2+3.5-17.2*C
238    M=I+4
```

כדי לעבור אל הוראה רצויה יש להשתמש בהוראת הסיעוף n GØ TØ.
כאשר n הוא מספר ההוראה שאליה מסתעפים. לאחר ה"סיעוף" אל ההוראה n ימשך ביצוע התוכנית מהוראה זו. זהו סיעוף בלתי מותנה, שלא כמו במשפטי ההתניה שבהם קיימת אפשרות ברירה.

כדי לעצור את ביצוע התוכנית יש להשתמש בהוראת העצירה STØP.
בהגיע התוכנית להוראה זו יופסק ביצועה והפיקוח יעבור אל תוכנית הניהול (מערכת ההפעלה).

בכתיבת תוכניות קיימות שתי הוראות נוספות, שאינן חלק מהתוכנית אלא מיועדות למהדיר: ההוראה END חייבת להיות האחרונה בכל תוכנית פורטרן. שלא כפקודת ה-STØP אשר גורמת להפסקת הביצוע של התוכנית, ל-END אין כל משמעות מבחינת הביצוע של התוכנית. היא מציינת למהדיר שזהו הסוף הפיזי של תוכנית המקור וכרטיסים שיבואו לאחר מכן אינם כרטיסי התוכנית. אלו יכולים להיות כרטיסי בקרה נוספים למחשב, נתונים לתוכנית, תוכנית אחרת וכו'.

בניבים מסויימים של פורטרן (למחשבי CDC) יש להקדים את התוכנית

בהוראה PRØGRAM.

המבנה של ההוראה הוא: PRØGRAM name (INPUT, ØUTPUT)

name הוא שם התוכנית, משתנה בן 7 תוים כאשר הראשון ביניהם הוא אות.

בסוגריים מציינים ØUTPUT, INPUT כאשר התוכנית נדרשת לבצע פעולות קלט ופלט.

לדוגמא:

PRØGRAM SEARCH (INPUT, ØUTPUT)

משפטי התנייה

משפט התניה מורה למחשב לבדוק ערכו של ביטוי או של משתנה ולהסתעף להוראות רצויות בהתאם לתוצאות הבדיקה.

קיימות צורות אחדות למשפטי התניה. המשפט הבסיסי הוא משפט "אם" IF statement.

משפטי התניה אריתמטיים

מבנה משפט התנאי האריתמטי הוא

$$IF\ (a)\ n_1,\ n_2,\ n_3$$

כאשר a הוא ביטוי אריתמטי ו-n_1, n_2, n_3 מספרי הוראות. תמיד יהיה סיעוף בביצוע הוראה זו והתוכנית <u>לא</u> תמשיך אוטומטית להוראה העוקבת. בזמן ביצוע הוראה זאת מחושב ערכו של הביטוי האריתמטי a ואז:

א. אם $a<0$ תבוצע ההוראה שמספרה n_1

ב. אם $a=0$ תבוצע ההוראה שמספרה n_2

ג. אם $a>0$ תבוצע ההוראה שמספרה n_3.

באמרנו "תבוצע ההוראה שמספרה n_1", למשל, כוונתנו היא שמסלול ביצוע התוכנית יעבור להוראה n_1 וימשיך מאותה נקודה ואילך.

<u>דוגמא:</u> IF (x-5.0) 18 ,3 ,29

פירושו

אם $x-5.0<0$ עבור להוראה מספר 18 והמשך משם את ביצוע התוכנית

אם $x-5.0=0$ עבור להוראה מספר 3 והמשך משם את ביצוע התוכנית

אם $x-5.0>0$ אבור להוראה מספר 29 והמשך משם את ביצוע התוכנית

נתרגל את נושא משפטי ההתניה בפתרון הבעיה הבאה:
יש לחלק את המספר TØTAL במספר CØUNT ולבדוק את התוצאה. אם המנה היא מספר שלם - יש להפסיק; ואם המנה כוללת ערך עשרוני - יש להוסיף את המשתנה B ל-CØUNT, לחלק ולבדוק שוב.

<u>שורה</u>

```
1          TØTAL=15.
2          CØUNT=4.
3          B=0.5
4     50   RESULT=TØTAL/CØUNT
5          N=RESULT
6          IF (N-RESULT)1,2,1
7     1    CØUNT=CØUNT+B
8          GØ TØ 50
9     2    STØP
10         END
```

הסבר לתוכנית:

בשורות 1, 2, 3 נקבעו ערכים התחלתיים לבעיה. תוצאת החילוק בשורה 4 תתן מספר ממשי (כולל חלק עשרוני). בשורה 5 מעבירים את המספר למשתנה שלם, וע"י כך יאבד החלק העשרוני שלו. הבדיקה IF בשורה 6 מכוונת להשוות את הערך של המשתנה לערכו של המשתנה RESULT. אם החלק העשרוני היה שונה מ-אפס לא יהיה שויון ויבוצע סיעוף להוראה מספר 1. אם החלק העשרוני שוה לאפס, השויון יתקיים ויבוצע סיעוף להוראה 2 (בשורה 9)והתוכנית תסתיים. בהוראה מספר 1 בשורה 7 מוסיפים ל-CØUNT את הערך B, ובשורה 8 מבצעים סיעוף בלתי מותנה אל ההוראה מספר 50 שבשורה 4.

נחזור לפתרון המשוואה הריבועית, ונוסיף בדיקה של הביטוי B**2-4.*A*C ושאר משפטי הבקרה הדרושים. התוכנית המשופרת תיראה כך:

```
1         A=2.
2         B=3.
3         C=-1.
4         DELSQ=B**2-4.*A*C
5         IF(DELSQ) 5,9,23
6     9   x1=-B/(2.*A)
7         x2=x1
8         GØ TØ 5
9    23   DEL=DELSQ**0.5
10        x1=(-B+DEL)/(2.*A)
11        x2=(-B-DEL)/(2.*A)
12    5   STØP
13        END
```

הסבר לתוכנית:

הביטוי שרוצים לבדוק הוא B**2-4.*A*C. מחשבים אותו בשורה 4 ומאחסנים אותו במשתנה בשם DELSQ. בשורה 5 בודקים את ערכו של משתנה זה.

אם הערך הוא אפס פירושו שלמשוואה שני פתרונות זהים הנתונים ע"י $x_{1,2}=-\frac{B}{2A}$ ולכן פונים להוראה מספר 9 בשורה 6 המחשבת את x1. כיוון ש-x2 שווה ל-x1 מבצעים את ההוראה בשורה 7 אשר מעבירה ל-x2 את ערכו של x1. לאחר חישוב שני הפתרונות פונים לפקודה מספר 5 (בשורה 12) ומסיימים.

אם ערכו של המשתנה $\sqrt{DELSQ}$ הוא חיובי, פונים להוראה מספר 23 (שורה 9) המחשבת את DEL אשר ערכו נתון ע"י DELSQ וממשיכם בחישוב x1 ו-x2. לאחר זאת מגיעים להוראה מספר 5 (בשורה 12) ועוצרים.

<u>הערה</u>: הקורא בודאי שם ליבו לעובדה שמספרי ההוראות אינם חייבים להופיע בסדר עולה. כל מספר (פרט ל-0) בעל 5 ספרות או פחות - מתקבל.

משפטי התניה לוגיים

משפט התנאי אותו הכרנו עד כה בודק את ערכו של ביטוי אריתמטי ביחס ל-0

(קטן, גדול או שווה), ופונה להוראה הבאה בהתאם לתוצאת ההשוואה. משפט התנאי הבא שנלמד קרוי "משפט התנאי הלוגי" (logical IF statement) והוא שונה מקודמו בכך שאינו בודק ביטוי אריתמטי אלא ביטוי לוגי. היינו, הוא בודק האם הביטוי המתאים הוא "נכון" או "לא-נכון", או אם יחס מסוים מתקיים.

צורתו של משפט זה היא: IF (logical expression) statement.
statement הוא הוראה שיש לבצעה אם הביטוי הלוגי שבתוך הסוגריים הוא "נכון", כלומר אם התנאי הכתוב בתוך הסוגריים מתמלא. לאחר ביצוע statement תמשיך התוכנית במשפט העוקב, אלא אם זהו משפט GØ TØ המורה על סיעוף.

אם הביטוי הלוגי "אינו נכון", כלומר התנאי אינו מתמלא - תופנה התוכנית להוראה העוקבת את משפט IF ולא תבצע את statement. ל-statement ישנם מספר כללי כתיבה: למשל, הוא אינו יכול להיות משפט תנאי.

קיימות מספר צורות של ביטויים לוגיים. להלן יוצגו הביטויים המשמשים להשואת יחסים של ערכים מספריים. ברשימה זו מייצגים A ו-B משתנים או ביטויים אריתמטיים או ערכים קבועים - פירושם של הביטויים רשום לצידם:

A.EQ.B	Equal	A שווה ל-B
A.NE.B	Not Equal	A לא-שווה ל-B
A.GT.B	Greater Than	A גדול מ-B
A.GE.B	Greater or Equal	A גדול או שווה ל-B
A.LT.B	Less Than	A קטן מ-B
A.LE.B	Less or Equal	A קטן או שווה ל-B

<u>הערה</u>: הנקודות (.) צריכות להיות צמודות לסימני היחס

נבהיר את השימוש בהתניה הלוגית בעזרת מספר דוגמאות:

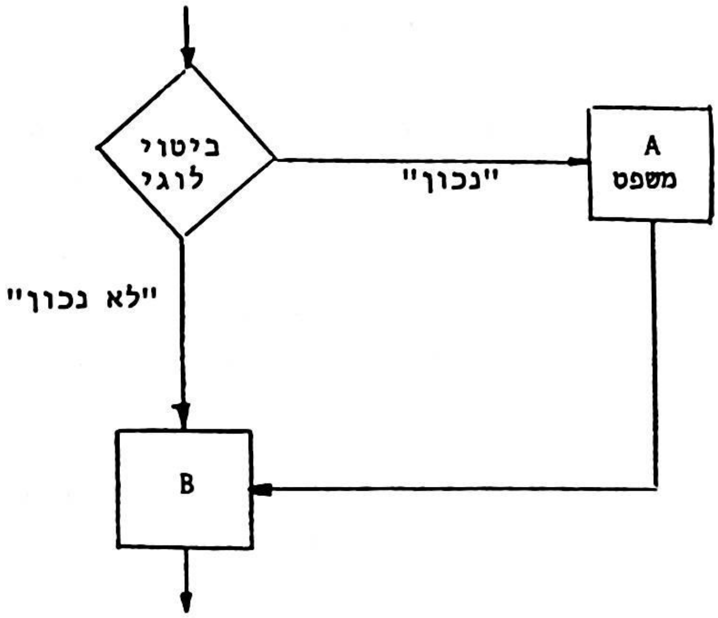
```
IF (X.GT.Y) GØ TØ 50
GØ TØ 70
```

בכניסה למשפט ה-IF יושוו הערכים של X ושל Y. אם X גדול (ממש) מ-Y (כלומר, אם הביטוי X.GT.Y הוא "נכון") תפנה התכנית לבצע את ההוראה שמספרה 50. ואילו אם המשפט "לא נכון" (היינו אם X קטן או שווה ל-Y) תופנה התוכנית להוראה מספר 70. דוגמא נוספת:

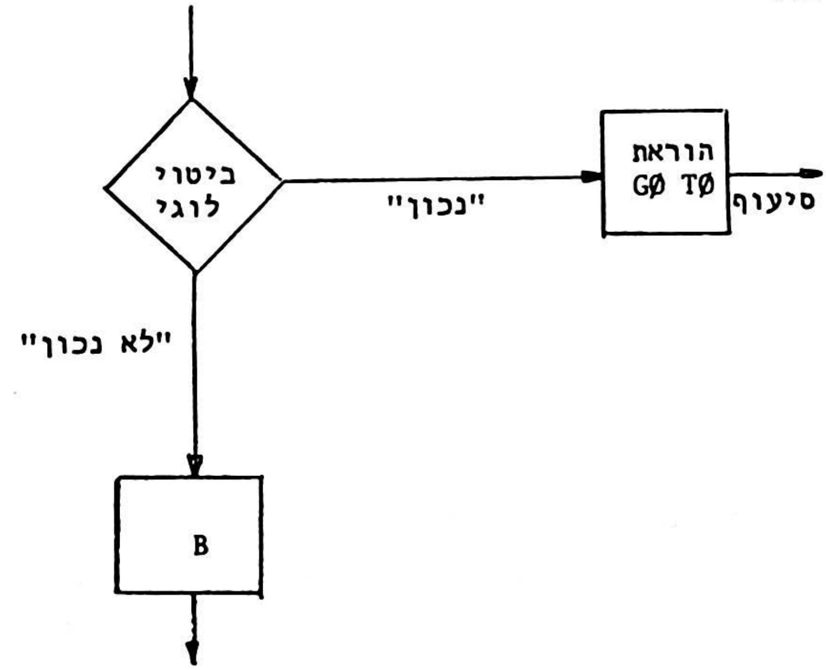
```
IF (I*J.LT.12) A=3.0
B=4.0
```

אם I*J קטן מ-12 אזי ינתן ל-A הערך 3.0, ולאחר מכן יבוצע המשפט B=4.0 ואילו אם ערכו של הביטוי I*J גדול או שווה ל-12 לא יבוצע המשפט A=3.0, אלא נפנה מיד למשפט העוקב, היינו, לביצוע המשפט B=4.0.

אפשר לסכם את האמור לעיל בתרשימים הבאים:



או:



אפשר לבנות ביטוי לוגי מצרוף של ביטויים הממלאים תנאים שונים. הבדיקה של הוראת IF תתיחס לביטוי הלוגי המורכב. צרוף של ביטויים יכול להעשות באמצעות אחת מהוראות הפעולה הבאות:

"לא" .NØT.

"גם" .AND.

"או" .ØR.

העדיפות לבצוע ההוראות היא בסדר כתיבתן לעיל. הנקודות (.) חייבות להופיע צמודות להוראות הפעולה.

דוגמא: אם A שוה ל-B וגם C גדול מ-D יש לבצע סיעוף להוראה 50.

```
IF (A.EQ.B.AND.C.GT.D) GØ TØ 50
```

כלומר, הסיעוף יתבצע רק כאשר שני התנאים מתמלאים. בצורה דומה אפשר לכתוב:

```
IF ((I*2.LT.5).ØR(B/2.3.NE.R)) GØ TØ 7
```

כאן יבוצע הסיעוף כאשר לפחות תנאי אחד יתמלא ("או"). הסוגריים החובקים כל ביטוי לוגי רצויים לשם בהירות אולם אינם הכרחיים.

עד עתה הצגנו את הוראת IF עם הוראת סיעוף אחת או משפט המתבצעים כאשר התנאי הוא "נכון". אם התנאי "לא-נכון" - מתבצעת ההוראה העוקבת.

קיימת אפשרות נוספת להצגת המשך בצוע ההוראות לאחר הוראת IF:

```
IF (logical expression) K,m
```

כאשר הביטוי הלוגי "נכון" - יבוצע סיעוף להוראה שמספרה K. אם הביטוי "לא נכון" - יבוצע סיעוף להוראה שמספרה m.

לדוגמא: IF (B.EQ.C) 7,25

הסיעוף להוראה 7 יבוצע כאשר B שוה ל-C.

הסיעוף להוראה 25 יבוצע כאשר B אינו שוה ל-C.

<u>טופס הקידוד</u>

דוגמאות התוכניות שהוצגו עד עתה נרשמו בצורה חופשית, ללא הקפדה על המבנה. כדי למסור את התוכנית למחשב יש לנקב את ההוראות בכרטיסים אשר משמשים קלט לתוכנית המהדיר (Compiler).

את ההוראות יש לנקב במבנה מוגדר ולשם כך משתמשים בטופס קידוד. הטופס מורכב משורות וטורים, בכל שורה נרשמת הוראה אחת בלבד, ובהתאם לכך ינוקב כרטיס אחד לכל שורה. בכל שורה 80 עמודות או טורים, המקבילים ל-80 הטורים של הכרטיס. כל תו (אות, ספרה או סימן) ירשמו בטור אחד, בנפרד. רווח בין תוים נוצר כאשר לא נרשם דבר והטור ריק או blank.

לכתיבת התוכנית בטופס הקידוד מספר כללים: הקלט למהדיר נרשם בטורים 1 - 72 וכולל את שדה מספר ההוראה, שדה סימון־המשך ושדה ההוראה:

<u>א</u>. הטורים 1 עד 5 נועדו לכתיבת "מספר ההוראה" או "תוית" - label. מספר זה הוא בעל 5 ספרות לכל היותר ורצוי לרשום אותו מוצמד לצד ימין של השדה.

<u>ב</u>. את ההוראות רושמים בטורים 7 עד 72. כתיבת ההוראות בטורים אלו היא חופשית למדי. המהדיר מתעלם ממרווחים בין המילים השונות היוצרות את ההוראה. כך למשל יכולה ההוראה A=B+C/3.0 להכתב כך, עם מרווחים:

A = B + C / 3 . 0

אסור לרשום מרווחים בין התוים של שם משתנה או מספר, למשל אסור MØ NE במקום MØNE. צורת הכתיבה תיבחר על ידי המתכנת לפי נוחיותו.

<u>ג</u>. לעתים ההוראה ארוכה יותר ממספר הטורים המותר ומעוניינים, או נאלצים, להמשיך את כתיבתה בשורה הבאה. כתיבת תו כלשהו בטור 6 (וניקובו) מורה למהדיר כי ההוראה הרשומה החל מטור 7 היא המשך של ההוראה בשורה הקודמת. בשורת ההמשך אסור לרשום דבר בטורים 1 עד 5. בשאר המקרים ישאר טור 6 ריק.

טורים 73-80 אינם מתורגמים על ידי המהדיר כהוראות פורטרן, אלא נועדו לנוחיותו של המתכנת. בדרך כלל רושמים בהם את המספר הסידורי של השורה ואפשר גם שם התוכנית. שים לב: אין הכוונה ל"מספר ההוראה".

לנוחיותו של האדם הקורא את התוכנית רצוי להוסיף הסברים בגוף התוכנית. כאשר רושמים C בטור 1 "מבין" המהדיר שלפניו שורת הסבר שיש להדפיסה בלבד אולם לא לתרגמה לשפת-מכונה.

פרק 2: פעולות קלט פלט והגדרת נתונים

כל התוכניות שהוצגו עד כה יכולות לפתור בעיות שאינן דורשות הזנת נתונים. בפתרון המשואה הרבועית, למשל, נקבעו ערכי המקדמים C,B,A בתוך התוכנית עצמה. גם בבעיות האחרות נקבעו הנתונים בתוכנית מראש. הנתונים היו יכולים להגיע לקטע התוכנית במהלך העיבוד אילו היתה תוכנית זו קטע מתוכנית גדולה יותר.

בבעית המשואה הריבועית עדיף היה לו יכולנו להורות למחשב לקרוא את ערכי המקדמים מתוך כרטיסים מנוקבים, למשל. פעולה זו היתה חוסכת לנו תקון התוכנית והידורה מחדש עבור כל סדרת מקדמים. בנוסף לכך אנו מעונינים שהמחשב ידפיס את פתרונות המשואה x1 ואת x2.

<u>הוראות לקריאה וכתיבה ומשפטי איפיון</u>

לשם ביצוע פעולות קריאה וכתיבה זקוקים לפקודות קלט ופלט. בפורטרן קיימות מספר פקודות למטרה זו. כאן נלמד רק שתיים מהן:

```
READ n, list
PRINT n, list
```

ההוראה READ מיועדת לקריאת נתונים מיחידת קלט. בצורתה הנוכחית, כל הוראה תגרום לקריאת כרטיס מנוקב אחד (או רשומה אחת של נתונים). אפשר לקרוא את <u>כל</u> 80 הטורים של הכרטיס ללא כל קשר לטופס הקידוד המיועד לכתיבת הוראות התוכנית.

ההוראה PRINT מיועדת לפלט נתונים בהדפסה. כל הוראה תגרום להדפסת שורה אחת במדפסת. רוחב שורת ההדפסה (מספר הטורים) בדרך כלל 132 עמודות. ישנו רוחב הדפסה שונה בהתאם למדפסת. את עמודות ההדפסה מונים מצד שמאל של השורה (עמודה מס' 1).

בשתי הפקודות הללו, n הוא מספר ההוראה (תוית) של משפט האיפיון ו-list

הוא רשימה של שמות המשתנים אותם מתכוונים לקרוא או לכתוב. המשתנים ברשימה מופרדים ע"י פסיקים. משפט האיפיון "מסביר" למחשב כיצד עליו לקרוא או להדפיס את המשתנים שב-list ומיד נשוב ונדון בו.

בתוכנית המשוואה הריבועית, למשל, יש לקרוא מתוך כרטיסים מנוקבים את A, B ו-C. ההוראה המתאימה לכך תהיה:

READ 100, A, B, C

ולהדפסת התוצאות x1 ו-x2 נכתוב:

PRINT 101, x1, x2

המספרים 100 ו-101 הם מספרי הוראות האיפיון המגדירות את שדות הקלט או הפלט בהתאמה. הצורה של הוראת האיפיון בפורטרן היא:

n FØRMAT $(S_1, S_2, S_3, \ldots, S_k)$

n הוא מספר הוראה המתאים למספר בהוראת ה-READ או ה-PRINT. S_1, S_2,.. הם האיפיונים השונים המגדירים את המשתנים.

בודאי זכור לקורא כי המשתנים יכולים להיות שלמים או ממשיים, ולכן חייבים להיות לפחות שני איפיונים שיאפשרו לנו לקרוא שני סוגים אלה. בנוסף לכך קיימים איפיונים לשדות אלפנומריים ולהגדרת רווח בין שדות. להלן נדון רק באיפיונים הבסיסיים. ההבדלים העיקריים בין האיפיונים לקריאה וכתיבה הם:

א. <u>רווחים בין שדות</u>: השדות ברשומת קלט בכרטיס צמודים בדרך-כלל, ללא רווח ביניהם. בהוראת האיפיון להדפסה יש לדאוג לרווחים בין השדות במכוון, ע"י איפיון המרווח (x) או ע"י ציון שדה הגדול ממספר הספרות שבמספר (המספר מוצמד לימין תחום השדה).

ב. <u>נקודה עשרונית</u>: בשדה של מספר ממשי ברשומת קלט יכולה, אך לא חייבת, להרשם הנקודה העשרונית. כאשר הנקודה אינה רשומה יוגדר השדה לפי האיפיון של אותו שדה (F). בהדפסה תודפס תמיד הנקודה העשרונית ולכן השדה צריך להכיל תו נוסף לצורך זה.

בהדפסה יש להבטיח שדה בגודל מספיק גם לסימן.

<u>איפיון קידום שורות במדפסת</u>

משפטי האיפיון מיועדים להגדרת נתונים: מבנה שדות בקלט ובפלט, כולל שדות להדפסה. אולם בהדפסה יש צורך לפקח על קידום הנייר כדי להדפיס את השורות לפי

תיכנון הדוח או הטופס המודפסים.

הוראות הבקרה העיקריות למדפסת הן קידום הנייר בשורה אחת, קידום הנייר בשתי שורות, או קידום הנייר לראשית הדף הבא.

ההוראה ניתנת ע"י תו-קוד באיפיון אלפנומרי בתחילת כל שורת הדפסה. בתחילה תבוצע הוראת הבקרה, ולאחריה תבוצע הדפסת השורה.

תוי הקוד הם: 1 לקידום הנייר לדף חדש
0 לקידום הנייר בשתי שורות
ריק לקידום הנייר בשורה אחת (קידום רגיל).

לדוגמא (הסבר למבנה האיפיון ראה להלן):

קידום לדף חדש:
```
5 FØRMAT (1H1,....)
```
או
```
5 FØRMAT (*1*,....)
```

קידום בשורה אחת:
```
8 FØRMAT (1H ,....)
```
או
```
8 FØRMAT (* *,....)
```

אפשר לחסוך את תו-הקוד לקידום שורה אחת אם האיפיון הראשון בשורה גדול מהשדה המספרי המוכל בו. במקרה זה יהיו תוי blank משמאל למספר, כנדרש להוראת הבקרה.

איפיון I למספרים שלימים

האיפיון שנועד להגדרת המשתנה השלם צורתו היא:

Iw

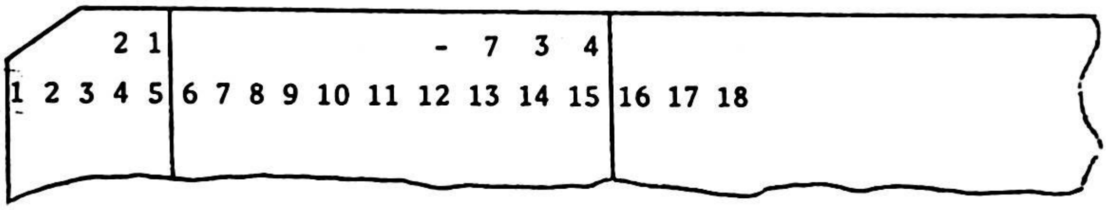
w הוא מספר שלם המביע את מספר הטורים (רוחב השדה) בכרטיס המנוקב או בדף המדפסת שיוקדשו למשתנה זה. המספר צריך להיות מספיק גדול בכדי להכיל גם הסימנים "+" או "-" אשר עשויים להופיע עם ערך המשתנה. למשל אם המספר הוא 123 - אז האיפיון I4 מתאים.

בקלט - I4 מציין שהנתון מנוקב על פני 4 טורים בכרטיס, מהם טור אחד מיועד לסימן ושלושה טורים לספרות. אפשר לקרוא גם את המספר 21 לפי אותו איפיון. במקרה כזה יש לדאוג שספרת היחידות של המספר (הספרה 1) תרשם בטור הימני של השדה.

להמחשה נוספת נניח שדרוש לקרוא שני מספרים I, J כאשר ל-I לכל היותר 4 ספרות (ואולי גם סימן), והשני של 9 ספרות לכל היותר (ואולי גם סימן). אם המספרים חיוביים אין צורך בהגדרת טור נוסף לסימן. המשתנה הראשון I ינוקב בחמישה הטורים הראשונים בכרטיס והשני בעשרה הטורים הבאים. ההוראה לקריאתם היא:

```
    READ 3,I,J
3   FØRMAT (I5, I10)
```

אם הנתונים הם I=21, J=-734 הכרטיס המנוקב צריך להיות:



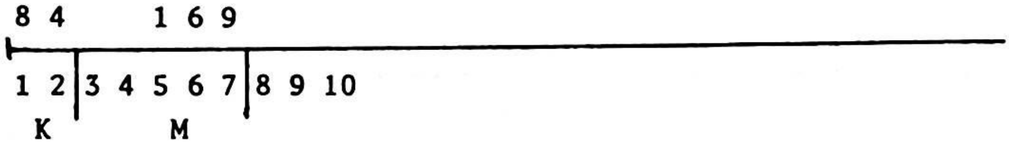
כיון ש-I הוא בעל שתי ספרות ואילו השדה המתאים הוא בן 5 טורים בכרטיס, יש להצמיד את המספר לצד הימני של השדה. כנ"ל לגבי המספר השני.

איפיון ההדפסה דומה במבנהו לאיפיון הקריאה. אם השדה גדול במכוון לעומת מספר שיודפס בו, יווצר רווח בין המספרים, כפי שנדרש בהוראות הדפסה.
לדוגמא: יש להדפיס את המספרים K=84 ו-M=169 עם רווח של 2 טורים ביניהם. הוראת האיפיון של השדה הראשון תהיה I2 ושל השדה השני - I5, כדי לכלול 2 טורי רווח.

מכיון שיש להשאיר תו ראשון בשורה כתו-קוד להוראת קידום הנייר, נרשום גם את האיפיון הראשון גדול מהנדרש עבור השדה:

```
    PRINT 9,K,M
9   FØRMAT (I3, I5)
```

שורת ההדפסה תהיה:



תו קוד לא מופיע בהדפסת שורה

<u>איפיון F למספרים ממשיים</u>

האיפיון המתאים למשתנה הממשי צורתו היא Fw.d.

w מציין את רוחב השדה (מספר הטורים בכרטיס שמוקדשים למשתנה, או מספר טורי

ההדפסה שמוקדשים לו), d מציין את מספר הספרות שמופיעות מימין לנקודה העשרונית.
המספר 173.2- למשל יוגדר עם האיפיון F6.1 שהרי זקוקים ל-6 טורים: 1 - עבור הסימן, 3 - עבור ספרות השלימים, 1 - לנקודה העשרונית, 1 - עבור הספרה שמימין לנקודה העשרונית. גם במקרה זה יכול מספר הספרות של המספר להיות קטן מהאיפיון אשר בעזרתו הוא נקרא או נכתב.

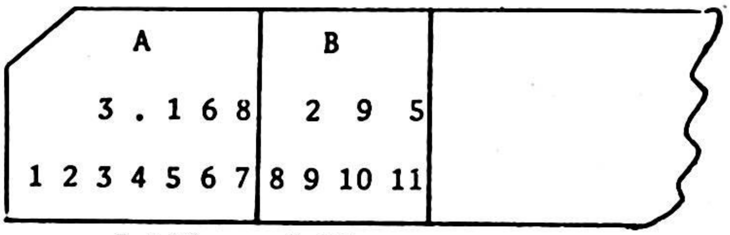
לדוגמא:

```
      READ 5,A,B
    5 FØRMAT (F7.3,F4.1)
```

לפי האיפיון מוקדשים למשתנה A 7 טורים ראשונים בכרטיס והוא מכיל 3 ספרות אחרי הנקודה העשרונית.

למשתנה B מוקדשים 4 טורים, עם ספרה עשרונית אחת.

המשתנה A ינוקב עם הנקודה העשרונית. המשתנה B ינוקב ללא הנקודה העשרונית.
בקלט - הנקודה העשרונית אינה חובה. בכל מקרה שהיא מופיעה בניקוב - היא גוברת על איפיון הפורמט המתאים לאותו שדה.



ערך המשתנה לפי האיפיון: 3.168 2.95

הוראת האיפיון להדפסה, דומה להוראת האיפיון לקריאה.

לדוגמא: האיפיונים להדפסת המספר 5.38- והמספר 98.7 הם:

F5.2 (5 גודל השדה, 2 ספרות מימין לנקודה העשרונית)

F4.1 (בהנחה שהמספר חיובי תמיד).

יש לזכור שמספר ממשי נשמר בזכרון המחשב ללא נקודה עשרונית אלא בשיטת הנקודה הצפה עם מנטיסה ומעריך חזקה. ולכן - איפיון השדה קובע את מבנה ההדפסה.

בזכרון המחשב מנוהל מספר ממשי עם ספרות רבות לאחר הנקודה העשרונית, בדרך כלל מעל הדרוש לנו. האיפיון קובע את מספר הספרות העשרוניות אשר יופיע בהדפסה.

לדוגמא: אם במחשב נשמר מספר חיובי שערכו הוא 57.83497 (למשל תוצאת חלוק או כפל) ורוצים להדפיסו רק עם שתי ספרות, יהיה איפיון ההדפסה F5.2 והשדה המודפס יהיה 57.83.

מרווח בין שדות מודפסים אפשרי ע"י קביעת איפיון גדול מגודל השדה הנדרש עבור המספר.

דוגמא להדפסת משתנה שלם ומשתנה ממשי:

```
   PRINT 15, NX, ABC
15 FØRMAT (I10, F12.4)
```

שני המשתנים יודפסו לפי סדר הופעתם באותה שורה. הראשון, NX, בעשרת הטורים הראשונים של השורה (משמאל) והמשתנה השני, ABC, ב-12 הטורים הבאים במבנה עם 4 ספרות מימין לנקודה העשרונית.

<u>איפיון X לציון מרווחים</u>

איפיון זה נועד לאפשר הגדרת מרווחים בין שדות המופיעים בהוראת האיפיון. והוא דרוש כאשר יש להדפיס מספרים עם מרווחים ביניהם. כגון:



צורתו הכללית של האיפיון: nX, כאשר n הוא מספר המרווחים שיש להשאיר.

נתון ש-K=13, V=19.35 ו-W=8.939 אז ההוראה

```
  PRINT 9,K,V,W
9 FØRMAT (4X, I2, 3X, F5.2, 5X, F5.3)
```

גרום להדפסת השורה הרשומה לעיל: 3 מרווחים (מרווח ראשון הינו תו קוד), מספר שלם, 3 מרווחים, מספר ממשי, 5 מרווחים ולבסוף מספר ממשי נוסף.

באופן דומה יבטא האיפיון X בהוראת קריאה את העובדה שברצוננו לדלג על קריאתם של מספר טורים בכרטיס.

אם הכרטיס הנקרא הוא:



אז ההוראה

```
   READ 10, I, Q
10 FØRMAT (2X, I1, 3X, F4.2)
```

תגרום לכך שהמשתנה I יקבל את הערך 9 (שהרי 2X בתחילת משפט ה-FØRMAT אומר לדלג על 2 הטורים הראשונים בכרטיס) ואילו Q יקבל את הערך 5.27 שהרי 3X אומר לדלג על שלושת הטורים 3, 4 ו-5.

איפיונים אלפנומריים A, H, *, '

עד עתה עסקנו באיפיונים מספריים ובאיפיון המרווח. בכתיבת תוכנית דרושים איפיונים להגדרת שדות אלפנומריים בקלט או בפלט.

האיפיון A, מיועד לבצוע פעולות קלט ופלט עם משתנים. האיפיונים H, *, ' (גרש) מיועדים בדרך כלל לבצוע פעולות פלט נתונים המוגדרים בהוראת האיפיון עצמה.

דוגמא לקריאת והדפסת נתונים אלפנומריים:

```
   READ 25, NAME, ADDR
25 FØRMAT (A16, A24)
```

הוראה זו תגרום לקריאת שני שדות אלפנומריים, האחד באורך 16 תוים והשני - באורך 24 תוים.

הוראת ההדפסה

```
   PRINT 30, NAME, ADDR
30 FØRMAT  (1X, A16, A24)
```

תגרום להדפסת שני שדות אלו.

שים לב לעובדה שבמחשבים שונים קיימות מגבלות לגבי האורך המותר לאיפיון שדה אלפנומרי.

```
  TØTAL = A*8.3
  PRINT 9, NAME, TØTAL
9 FØRMAT (15H 1 T H I S  I S  Ø U T P U T, 2X, A8, 2X, F7.2)
```

ההדפסה תבוצע בדף חדש ותראה כך:

```
(תו-הקו  T H I S   I S   O U T P U T     A A A A A A A A     N N N N . . N N
לא מופיע |<-           14              ->| |<-    8    ->| |<-      7      ->|
בשורה)      שדה אלפנומרי מהוראת            שדה אלפנומרי       שדה נומרי
                   האיפיון
```

האיפיון H מציין שבהוראת האיפיון כלולים תוים אלפנומריים והם נמצאים מימין לאות H. במקרה זה סה"כ - 15 תוים כפי שרשום בהוראה (המספר שלפני ה-H). התו הראשון הוא תו קוד למעבר לדף חדש. במחשבים מסוימים אפשר לכלול תוים אלפנו-מריים בהוראת האיפיון ע"י כתיבתם בין שני כוכבונים (*):

```
9 FØRMAT (*1THIS IS ØUTPUT*, 2X, A8, 2X, F7.2)
```

אפשר גם לכתוב איפיון הדפסה בצורה זו, כדי לכלול הסבר לנתונים המודפ-סים:

```
777 FØRMAT (1HO, 4X, 13H S T U D E N T  N A M E :, A20, 2X, 4H A G E :I2)
                          13                                     4
```

או

```
888 FØRMAT (*0*, *TØTAL:*, F8.3, * AVERAGE:*, F7.3, 2X, I4)
```

כפולות של איפיוני שדה

במקרים רבים נמצאים בקלט או בפלט מספר שדות רצופים בעלי מבנה זהה. אפשר לייעל מעט את כתיבת הוראת האיפיון ע"י שימוש בגורם הכפל.

יש לקרוא מכרטיס 3 מספרים שלימים, כל אחד בשדה באורך של 5 טורים, ושישה מספרים ממשיים כל אחד משדה באורך 10 טורים עם 4 ספרות אחרי הנקודה העשרונית. נוכל לבצע זאת בשימוש בהוראת האיפיון הבאה:

```
99 FØRMAT (3I5, 6F10.4)
```

כלומר, עם גורם כפל 3 של השדה I5, וגורם כפל 6 של השדה F10.4

עתה נרצה להדפיס שלשה משתנים, כאשר בין אחד לשני יש רווח של תו אחד

```
XX.XX  XX.XX  XX.XX
```

הוראת האיפיון להדפסה תהיה:

```
999 FØRMAT (F5.2, 1X, F5.2, 1X, F5.2)
```

הוראה זו אפשר לכתוב כך:

```
999 FØRMAT (3(F5.2, 1X))
```

כלומר, חזרה 3 פעמים על האיפיון של המשתנה ואיפיון המרווח שלאחריו.

תרגיל:

נחזור פעם נוספת לתוכנית המשוואה הריבועית והפעם נרשום אותה בצורתה הסופית: בשורה הראשונה מבקשים לקרוא את המשתנים C,B,A מכרטיס מנוקב לפי הוראת איפיון מספר 100. אם נסתכל בהוראה זאת נראה שמקדישים עשרה טורים לכל אחד מהמשתנים ומרשים דיוק של 3 ספרות אחרי הנקודה העשרונית. כאשר $\sqrt{B^2-4AC>0}$ מחשבים את הפתרונות ומדפיסים אותם בפקודה מספר 10 לפי הוראת איפיון מספר 101. מדפיסים אותם בשדות של 15 טורים עם 4 ספרות אחרי הנקודה העשרונית.

בתוכנית זו נוסיף גם הערות להבהרת הפעולות ותאור מילולי של השדות המוד-פסים.

ראה את התוכנית בדף הבא:

IBM FORTRAN Coding Form

תוכנית לחישוב שורשים של משוואה ריבועית

Statement No.	FORTRAN Statement		Identification Sequence
	PROGRAM CALCUL(INPUT,OUTPUT)		CALC 010
C	THIS PROGRAM WILL RESOLVE X1 AND X2	כרטיס הערה	CALC 020
C	OF THE EQUATION $AX^2+BX+C=0$	כרטיס הערה	CALC 030
	READ 100,A,B,C	קריאת כרטיס הנתונים לפי הוראת אפיון 100	CALC 040
	DELSQ = B**2 - 4.*A*C		CALC 050
	IF (DELSQ) 5,9,23	בדיקת הערך שמתחת לסימן השורש	CALC 060
9	X1 = -B/(2.*A)	תוצאה כאשר הערך שמתחת לסימן	CALC 070
	X2 = X1	השורש שווה ל-אפס	CALC 080
	GO TO 10	סיעוף אל הוראה 10	CALC 090
23	DEL = DELSQ**0.5	תוצאה כאשר ישנם שני שורשים שונים	CALC 100
	X1 = (-B+DEL)/(2.*A)		CALC 110
	X2 = (-B-DEL)/(2.*A)		CALC 120
10	PRINT 101,X1,X2	הדפסת התוצאה לפי הוראת אפיון 101	CALC 130
5	STOP	סוף התוכנית, גם כאשר אי אפשר לחשב שורשים	CALC 140
100	FORMAT (F10.3,F10.3,F10.3)	הוראת אפיון לקריאה. אפשר גם (3F10.3)	CALC 150
101	FORMAT (F15.4,F15.4)	הוראת אפיון לכתיבה. אפשר גם (2F15.4)	CALC 160
	END	סוף כרטיסי התוכנית.	CALC 170

<u>תרגיל:</u>

נתונה כרטסת המכילה פרטים אישיים של סטודנטים. עבור כל סטודנט מנוקב כרטיס אחד. בטור 5 של הכרטיס ישנו סימון זכר/נקבה: 1 עבור סטודנטים, 2 עבור סטודנטיות. הכרטיס האחרון בכרטסת מכיל את הסימון 9 בטור 5. יש לכתוב תוכנית לספירת מספר הסטודנטים ומספר הסטודנטיות, להדפיס את שני המספרים ואת סה"כ הסטודנטים.

התוכנית:

```
1        ICTR=0
2        MEN=0
3        IWMN=0
4     5  READ 4, ISEX
5     4  FØRMAT  (4X,I1)
6        IF (ISEX.EQ.9) GØ TØ 25
7        ICTR=ICTR+1
8        IF (ISEX.EQ.1) MEN=MEN+1
9        IF (ISEX.EQ.2) IWMN=IWMN+1
10       GØ TØ 5
11   25  PRINT 6, ICTR, MEN, IWMN
12    6  FØRMAT (1HO, 6HTØTAL=,I3,2X, 4HMEN=,I3,2X, 6HWØMEN=,I3)
13       STØP
14       END
```

הסבר:

שורות 1-3 - איפוס מונים: כללי, סטודנטים, סטודנטיות

5-4 - קריאת כרטיס. קוראים רק את המספר מטור 5 של הכרטיס ועל כן יש צורך לדלג על 4 הטורים הראשונים (4X). מכיון שהמספר הוא שלם מעבירים אותו למשתנה המייצג מספר שלם.

6 - בדיקה לכרטיס אחרון (9 במשתנה ISEX). אם הגענו לכרטיס אחרון עוברים לפעולות הסיום של התוכנית בהוראה 25 (שורה 11).

7 - ספירה כללית של סטודנטים וסטודנטיות: הוספת 1 למונה.

10-8 - ספירה נפרדת של סטודנטים וסטודנטיות לאחר בדיקת הסימון בשדה ISEX (טור 5 של הכרטיס). לאחר הבדיקה וההשואה מסתעפים להוראה 5 בשורה 4 - קריאת כרטיס נוסף

בשורה 8 בודקים את הסימון 1 וגם אם התשובה חיובית עוברים לבדי-
קת סימון 2 (בשורה 9) למרות שאין לכך כל משמעות, וגם לא נזק
או שגיאה. עשינו זאת לשם הפשטות. למעשה אפשר לעבור לאחר תשובה
חיובית בשורה 8 (כלומר הכרטיס הוא של סטודנט) לקריאת כרטיס חדש.

12-11 - מדפיסים את התוצאות בשורה אחת עם הסבר לכל ערך מודפס: השדה ICTR
הוא TØTAL - סה"כ הסטודנטים. השדות MEN ו-IWMN הם MEN ו-WØMEN
בהתאמה. שים לב שבעוד אשר בתוכנית הופיעו המשתנים כמייצגים
מספרים שלימים, אין לכך כל חשיבות כאשר מדפיסים את התוצאה עם
הסבר.

מבנה שורת ההדפסה:

TØTAL=XXX,--MEN=XXX,--WØMEN=XXX

השורה הזו תודפס ברווח של שתי שורות מקודמתה על פי קוד דילוג
השורות 0 (אפס - 1H0) אשר מופיע בראש השורה. מספר התוים אשר
יודפסו כהסבר לכל ערך מבוקר ע"י איפיון H - וזהו המספר המופיע
משמאל לאות H. שים לב לעובדה שאם רוצים להשאיר רווחים צריך
לדאוג לכך ע"י תוי רווח (blank) או ע"י האיפיון X.

קידום רשומות

הוראת האיפיון מגדירה בדרך כלל מבנה של רשומה פיזית אחת: את השדות המנו-קבים בכרטיס אחד או השדות המודפסים בשורה אחת. כאשר רשימת המשתנים בהוראת קלט/פלט ארוכה ממספר השדות המוגדרים בהוראת האיפיון תבוצענה אוטומטית פעולות קלט/פלט נוספות לפי אותה הוראת איפיון עד להשלמת רשימת המשתנים.

אפשר להגדיר בהוראת האיפיון פעולה אשר תגרום הפסקת הטיפול ברשומה הנוכחית והתחלת הטיפול ברשומה הבאה. כלומר השדות הבאים המוגדרים בהוראת האיפיון יודפסו בשורה חדשה או יקראו מכרטיס חדש.

הוראת הקידום היא הקו הנטוי (/).

לדוגמא:

```
    A=273.7
    B= 84.9
    PRINT 100,A,B
100 FØRMAT (*1 HEADING*,//,F8.1,/,F8.1)
```

צורת הפלט הנובעת מהוראת ההדפסה עם הוראת האיפיון 100 תהיה:

שורת כותרת בראש הדף (קידום נייר = 1):	HEADING
לאחר קידום של שתי שורות:	273.7
לאחר קידום של שורה אחת:	84.9

הוראות עם הכרזת יחידת קלט/פלט

עד עתה הצגנו פעולות קלט מכרטיסים ופלט במדפסת, בהנחה שיחידות אלו מוג-דרות בקביעות כיחידות הסטנדרטיות של המערכת. אולם, במערכת המחשב ישנן יחידות קלט ופלט נוספות מסוג סרטים מגנטיים, דיסקים מגנטיים, מנקבת כרטיסים, מכונת כתיבה ויתכן גם קוראת כרטיסים ומדפסת נוספת. בהוראות READ ו-PRINT שהוגדרו אין כל התיחסות למגוון יחידות אלו.

כדי להפעיל יחידות קלט ופלט שונות מהסטנדרטיות יש להכריז עליהן בהוראת READ לקריאה ובהוראת WRITE לכתיבה, בצורה הבאה:

```
READ (unit, format) variable list
WRITE (unit, format) variable list
```

unit - מספר שלם או משתנה שלם המציין את ההגדרה של היחידה במחשב.
format - מספר הוראת האיפיון.
variable list - רשימת המשתנים שאותם רוצים לקרוא או לכתוב.

ההגדרה המקובלת של יחידות קלט ופלט היא:

5 - קוראת כרטיסים (במחשבי י.ב.מ.)
6 - מדפסת (" ")
7 - מנקבת כרטיסים (" ")
10-99 - יחידות קלט ופלט שונות.
כל מספר מתיחס ליחידה מסוימת כפי שהוגדרה במחשב.

בדרך כלל ההגדרות הן סטנדרטיות כדי להקל על הפעלת התוכנית בכל מחשב.

כדי שהתוכנית לא תהיה תלויה ביחידת קלט ופלט שהוגדרה בה ובהנחה שהוראות קלט ופלט נרשמות בתוכנית פעמים רבות אפשר להשתמש בשיטת ההגדרה הלוגית ליחידות ע"י משתנה:

```
READ (IN1,999)A,B,C
```

בתחילת התוכנית תרשם ההוראה המקנה ערך למשתנה IN1:

IN1 = 5 יגדיר יחידת קלט - קוראת כרטיסים,
או: IN1 = 29 כדי להגדיר יחידת קלט - סרט מגנטי מסוים.

לסרטים מגנטיים יש הוראות מיוחדות אופייניות:

REWIND unit - גלגול הסרט לתחילתו
BACKSPACE unit - החזרת הסרט ברשומה אחת
ENDFILE unit - כתיבת סימון "סוף קובץ"

הכרזת נתונים - Data

משתנים מקבלים ערכים בצורות אחדות: ע"י הוראת השמה של מספר קבוע, ע"י הוראת השמה של תוצאת ביטוי אריתמטי או ע"י פעולת קלט. המשותף לצורות אלו היא העובדה שמתן הערך למשתנה נעשה בזמן בצוע התוכנית. אפשר להכין מראש, בזמן הידור התוכנית, ערכים במשתנים, וזאת באמצעות ההוראה Data. ערכים אלו יכולים להשתנות במהלך התוכנית.

את ההוראה Data לאחר הכרזות אחרות המופיעות בראש התוכנית, ולפני הוראות הבצוע.

מבנה ההוראה:

```
DATA PI/3.14/
DATA I,A,X/7,2.5,80.3/
```

Data הוראת ההכרזה

- רשימת משתנים מופרדת בסימן פסיק (,)
- קו נטוי
- רשימת ערכים מופרדת בסימן פסיק. סדר הערכים וסוגם צריך להתאים לרשימת הערכים.
- קו נטוי

להכנת מערך, אפשר להשתמש בהוראה הבאה:

```
DIMENSIØN  M(7)
DATA (M(I),I=1,7)/9,6*2/
```

האיבר הראשון יקבל את המספר 9 וכל 6 האיברים הבאים - את המספר 2.

פרק 3: מערכים ולולאות

מערכים - וקטורים ומטריצות

בבעיות מדעיות יש לעסוק, לעיתים קרובות, במערכים של מספרים (וקטורים, מטריצות, טבלאות) ועל כן שפת התכנות צריכה "להבין" מושגים אלה. באופן פשטני אפשר להגדיר מערך כאוסף של מספרים.

מבנה המערך

המערך הראשוני הוא מערך חד-ממדי או הוקטור. נתון, למשל, אוסף המספרים הבא:

15	17	93	21	30
a_1	a_2	a_3	a_4	a_5

בכתיב המתימטי נסמן את הוקטור באות a, למשל, ולאיברים של הוקטור נפנה ע"י הוספת אינדקס לשם הוקטור. כך, למשל, יסמן a_1 את המספר 15 בדוגמא לעיל, a_3 את 93 וכדי'. סימון דומה לוקטור קיים גם בפורטרן.

שם הוקטור נקבע לפי הכללים של בחירת שם למשתנה. אם הוקטור הוא וקטור של מספרים שלימים - השם חייב להתחיל באחת האותיות I,J,K,L,M,N; ואם הוא וקטור של מספרים ממשיים - יתחיל השם בכל אות אחרת מאלו שצויינו. האינדקס לציון האיבר המסויים בוקטור יירשם בתוך סוגריים הצמודים לשם הוקטור. כך A(1) יציין את האיבר הראשון בוקטור A, A(5) יציין את האיבר החמישי וכו'. האינדקס של וקטור איננו חייב להיות קבוע: A(I) יציין את האיבר ה-I בוקטור.

דוגמא מורכבת יותר היא המערך הדו-מימדי הקרוי גם בשם מטריצה או טבלה. למשל:

9	12	1
4	3	5
7	2	8

המטריצה מורכבת ממספרים הערוכים בשורות ועמודות. גם הפעם משתמשים בשם, למשל B, כדי לציין את המערך כולו. וכיצד נפנה לאיבר מסויים במטריצה? B(2,3) מציין את המספר הנמצא בשורה השניה ובעמודה השלישית של המטריצה , כלומר, המספר 5 בדוגמא שלפנינו. ובאופן כללי יציין B(I,J) את האיבר של המטריצה B אשר נמצא בשורה ה-I ובעמודה ה-J.

Dimension הוראה

כדי שנוכל להשתמש במערכים השונים, יש להורות למהדיר מיהם המערכים - לפי שמותיהם וממדיהם. ההכרזה המוקדמת של ממדי המערך נובעת מן העובדה שעל המהדיר לשמור מראש, בזמן בנית התוכנית, מספר מספיק של תאי זכרון עוקבים עבור כל המערך. משתנים רגילים, אינם חיובים להיות צמודים זה לזה.

ההכרזה על מימדי מערך נעשית ע"י ההוראה הבאה:

```
DIMENSIØN u, v, w
```

כאשר u,v,w הם השמות של המערכים: כל מערך רשום עם המספר המקסימלי של האיברים הכלולים בו. כך, למשל, ההוראה

```
DIMENSIØN A(100)
```

מכריזה כי A הוא מערך חד-מימדי הכולל לכל היותר 100 איברים. וההוראה DIMENSIØN A(4), B(7,4) מכריזה כי A הוא וקטור של 4 איברים ואילו B היא מטריצה בעלת 7 שורות ו-4 עמודות (סה"כ 28 איברים).

נשים לב לכך שמספר האינדקסים של מערך בהוראת ה-DIMENSIØN חייב להיות זהה למספר האינדקסים בזמן השימוש במערך

אם הוראת ה-DIMENSIØN היא: DIMENSIØN A(23)

אסור לנסות ולפנות לאיבר A(2,4) שהרי A הוגדר כוקטור והפניה אליו היא כאל איבר של מטריצה. כמו כן מובן שאסור לחרוג מתחומי הגדרתו של המערך. אסור לפנות אל A(24) כיון שהוראת ה-DIMENSIØN מגדירה את A כוקטור בעל 23 איברים לכל היותר.

פעולות אפשריות עם מערכים הן:

```
A(5) = B(8) + C(I,K)
A(I) = A(I) + 1
IF (MØNE (I).EQ.7) GØ TØ 80
```

ל ו ל א ו ת

לעתים יש צורך לבצע באופן מחזורי סדרת הוראות. למשל, בסיכום של סדרת מספרים יש לחזור על פעולת החיבור עבור כל אחד מהמספרים עד לסיום הסדרה.

כאשר יש לחשב ממוצע ציונים של תלמידים, יש לקרוא את הרשומות (כרטיסים)

של כל התלמידים. בכל רשומה נמצאים נתוני הציונים של כל תלמיד ועל פיהם מחושב הממוצע. פעולות הקריאה, החישוב וההדפסה של התוצאות תחזורנה עד לסיום כל קובץ הנתונים.

נלמד עתה איך לכתוב תוכנית אשר בה נדרשת חזרה על סדרת הוראות. ההוראות יכולות להיות קבועות ואחידות עבור כל הרשומות המעובדות, או שאפשר להתנות את ביצוע כולן או חלקן בהתאם לנתונים משתנים המתקבלים בפעולת הקריאה, או תוך כדי ביצוע ההוראות.

"הלולאה" (loop) הינה חזרה על סדרת הוראות במחזוריות. הבקרה של הלולאה, או מספר הפעמים שחוזרים על סדרת ההוראות, נקבע ע"י אינדקס או תנאי אחר כפי שיוסבר בהמשך.

<u>תכנות בעיה</u>

נפתור בעיה פשוטה בעזרת נוהל הלולאה:
נתון מערך M בן 100 איברים. יש להכניס בו את המספרים 1 עד 100 בהתאמה.

<u>שורה</u>

```
1      DIMENSIØN M(100)
2      I=1
3    4 M(I)=I
4      IF(I-100)1,2,2
5    1 I=I+1
6      GØ TØ 4
7    2 STØP
8      END
```

בבעיה זו נדרש מערך ועל כן נכריז עליו בעזרת ההוראה DIMENSIØN המגדירה מערך M בן 100 איברים המיועד להכיל מספרים שלימים. בשורה 2 נקבע ערכו ההתח-לתי של האינדקס ל-1. אינדקס זה יהיה המספר הסידורי של האיבר בטבלה. נשתמש בו גם כערך המספרי המוכנס לאיברי הטבלה.

בשורה 3 מעבירים את הערך של האינדקס אל האיבר המתאים במערך. בפעם הראשונה יהיה M(1)=1 ובפעם השניה: M(2)=2 וכן הלאה עד M(100)=100. בשורה 4

נבדק ערכו של האינדקס. אם I קטן מ-100 (זאת אומרת שהביטוי שבסוגריים הוא שלילי) יש להמשיך בקידום האינדקס (בשורה 5) ולחזור לתהליך בלולאה (משורה 6 אל ההוראה 4 בשורה 3). אם I שווה ל-100 יש לסיים את התוכנית לפי ההוראה בשורה 7.

בקביעת הערך להשוואה בהוראה IF (שורה 4) יש לשים לב אם קידום האינדקס והפעולה (לפי שורה 3) מתבצעים לפני או אחרי הוראת IF.

הבה ונפתור בעיה נוספת: חישוב ממוצע וסטיית תקן של סדרה של מספרים ממשיים. אם נניח שהסדרה היא בת 172 אלמנטים אז הנוסחאות המתאימות הן:

$$\bar{a} = \frac{1}{172} \sum_{i=1}^{172} a_i$$

$$\sigma^2 = \frac{1}{172} \sum_{i=1}^{172} a_i^2 - \bar{a}^2$$

כאשר a_i.$i=1,...,172$ הם המספרים הנתונים, $\bar{a}$ הוא הממוצע ו-σ^2 הוא הריבוע של סטיית התקן. המספרים a_i מנוקבים כל אחד בכרטיס בפורמט F10.3. נכתוב תחילה את התוכנית ולאחר מכן נסביר אותה.

IBM — FORTRAN Coding Form

תוכנית לחישוב ממוצע וסטית תקן של סדרת מספרים הנתונה בכרטיסים

FORTRAN STATEMENT	הערה	IDENTIFICATION SEQUENCE
`C PROGRAM TO CALCULATE AVERAGE AND STANDARD DEVIATION`		AVST 010
`C OF SERIES OF REAL NUMBERS.`		AVST 020
`SUM=0.`	פעולות הכנה	AVST 030
`SUMSQ=0.`		AVST 040
`I=0`		AVST 050
`5 READ 100,A`	קריאת כרטיסי נתונים	AVST 060
`SUM=SUM+A`		AVST 070
`SUMSQ = SUMSQ + A**2`		AVST 080
`I=I+1`		AVST 090
`IF ( I-172 ) 5,10,10`	בדיקה לסיום קריאת הנתונים (172 כרטיסים)	AVST 100
`10 AVER = SUM / 172.`		
`SIGSQ = SUMSQ / 172.-AVER**2`	פעולות עיבוד לאחר סיום	AVST 110
`SIGMA = SIGSQ**0.5`	קריאת כל כרטיסי הקלט	AVST 120
		AVST 130
`PRINT 200,AVER,SIGMA`	הדפסת התוצאה	
`STOP`		AVST 140
	סיום ביצוע התוכנית	AVST 150
`100 FORMAT ( F10.3 )`		
`200 FORMAT ( 2F10.3 )`		AVST 160
`END`		AVST 170
	סוף כרטיסי התוכנית	AVST 180

*A standard card form, IBM electro 888157, is available for punching statements from this form

המשתנים SUM ו-SUMSQ יכללו לקראת סוף ביצוע התוכנית את סכום המספרים ואת סכום הריבועים שלהם. מאפסים מונים אלו בתחילת התוכנית (שורות 1, 2) ובכל פעם שקוראים מספר a_i מוסיפים אליהם את המחובר המתאים. המשתנה השלם I נועד לספירת מספר ה-a_i-ים שכבר קראנו וגם אותו מאפסים בתחילת התוכנית.

הוראה מספר 5 קוראת את המספר a_i (לפי פורמט 100). לאחר שקראנו מספר, מוסיפים את ערכו ל-SUM (שורה 5) ואת ריבועו ל-SUMSQ (שורה 6), ולבסוף מוסיפים 1 למונה I (שורה 7). לאחר ביצוע ההוראה I=I+1 כולל I את מספר המשתנים שקראנו. כעת בודקים את ערכו של הביטוי I-172 (בשורה 8). אם הוא קטן מ-0 (כלומר: I קטן מ-172) יש עדיין משתנים שלא קראנו ולכן פונים שוב להוראה מספר 5 כדי לקרוא את המספר הבא. אם I-172 שווה ל-0 או גדול ממנו הרי שקראנו כבר את כל 172 המספרים ואפשר לפנות לחישוב הממוצע וסטיית התקן. הדבר נעשה בהוראה מספר 10 (בשורה 9) וההוראות הבאות אחריה. לאחר קריאת כל המספרים a_i, כולל SUM את סכום המספרים הללו. הממוצע AVER מחושב בפקודה מספר 10 על ידי חלוקת SUM ב-172.0. ממוצע הריבוע מתקבל מחלוקת SUMSQ ב-172.0. ריבוע סטיית התקן SIGSQ מחושב (בשורה 10) על-ידי

SIGSQ = SUMSQ/172.-AVER**2

לבסוף, מקבלים את סטיית התקן SIGMA על ידי חישוב השורש של SIGSQ (בשורה 11) ומדפיסים את הממוצע AVER ואת סטיית התקן SIGMA בפורמט 200 (בשורה 12).

נסתכל שנית על ההוראות שבין הוראה מספר 5 והוראה מספר 10; את ההוראות

SUM = SUM + A

SUMSQ = SUMSQ + A**2

זרים ומבצעים עבור כל משתנה A שקראנו.

מצב זה בו מבצעים מספר פעמים אותה סדרה של הוראות הוא ה"לולאה". הבקרה של הלולאה (קביעת הרגע שבו יש להפסיקה) נתונה על ידי ערכו של המשתנה I (האינדקס של הלולאה) הסופר את מספר הפעמים שהלולאה בוצעה. כיון שלולאות מופיעות בתוכניות מספר רב של פעמים קיימת בשפה הוראה מיוחדת - DØ המפקחת על ביצוע לולאות.

הוראה ביצוע לולאות - DØ

ההוראה DØ מאפשרת למתכנת לבצע סדרה של פעולות חוזרות בלולאה ולפקח על ביצוע זה עד לסיומו. הפיקוח על ביצוע הלולאה מתאפשר ע"י שימוש באינדקס אשר

משתנה בין שני ערכים קבועים מראש.

המבנה הכללי של ההוראה DØ:

DØ K i = n_1, n_2, j

ההוראה DØ נרשמת בראש סדרת ההוראות של הלולאה.

K - המספר (תוית) של ההוראה האחרונה באותה לולאה.

i - מייצג שם של משתנה שלם - האינדקס.

n_1, n_2 - קבועים או משתנים שלימים - תחום ההשתנות של האינדקס.

j - קידום האינדקס, לא חובה לרשום. (ברירת המחדל היא קידום של 1).

משמעות הוראה DØ היא: בצע בלולאה את כל ההוראות החל מההוראה העוקבת את ההוראה DØ ועד הוראה K ועד בכלל, כאשר האינדקס של הלולאה הוא i והוא משתנה מ-n_1 ועד n_2.

בפעם הראשונה שהלולאה מתבצעת מקבל i את הערך n_1 ובכל ביצוע נוסף מתווסף 1 ל-i עד אשר מגיע ערכו ל-n_2. לאחר מכן נפסק ביצוע הלולאה והבקרה עוברת להוראה אשר אחרי ההוראה שמספרה

אפשר ומותר להשתמש ב-I גם כדי לבצע חישובים התלויים ב-I. למשל, אם מעונינים לחשב את סכום הריבועים של 100 המספרים השלימים הראשונים נוכל לעשות זאת כך:

```
  MØNE = 0
  DØ 5 I=1,100
5 MØNE = MØNE + I**2
```

דוגמא נוספת:

נכתוב מחדש, בעזרת ההוראה DØ, את התוכנית להכנסת 100 מספרים (מ-1 עד 100) במערך בן 100 איברים.

```
  DIMENSIØN  M(100)
  DØ 5 I=1,100
5 M(I) = I
  STØP
  END
```

ההוראה המרכזית בתוכנית היא M(I)=I. בהוראה זו מעבירים ערך I, משתנה, אל האיבר המתאים בטבלה. בתוכנית זו לא צריך לטפל בקידום האינדקס כפי שנעשה בתוכנית הקודמת מכיון שהוא מבוקר, אוטומטית, בקידום של 1 ע"י ההוראה DØ.

הערך ההתחלתי שלו הוא 1 והערך האחרון - 100.

התוכנית שכתבנו לחישוב הממוצע וסטיית התקן יכולה להכתב גם בצורה הבאה:

```
      SUM=0.
      SUMSQ=0.
      DØ 10 I=1,172
      READ 100, A
      SUM=SUM+A
   10 SUMSQ=SUMSQ+A**2
      AVER=SUM/172.
      SIGSQ=SUMSQ/172.-AVER**2
      SIGMA=SIGSQ**0.5
      PRINT 200, AVER, SIGMA
      STØP
  100 FØRMAT (F10.3)
  200 FØRMAT (F10.3, F10.3)
      END
```

בתוכנית זאת משמש I רק לבקרת הלולאה. ערכו של I קובע אם להמשיך בביצוע הלולאה ואם לאו.

הוראת ההמשך CONTINUE

לעתים אי אפשר לסיים את הלולאה בהוראת ביצוע, מכיון שהיא אינה מתבצעת תמיד. מקרה זה קיים כאשר בתוכנית ישנם משפטי התניה המובילים אל סוף סדרת ההוראות בלולאה, ואינם צריכים לבצע כל הוראה שהיא, אלא לקבל הפניה אל תחילת הלולאה.

לצורך זה קיימת הוראה CONTINUE - הוראת המשך - אשר נרשמת בסוף הלולאה ונושאת את מספר ההוראה המתאים. היא אינה מבצעת דבר אלא גורמת להחזרת הפיקוח אל תחילת הלולאה, אם האינדקס טרם הגיע לגבול העליון, או אל הפקודה הבאה אם תנאי זה התמלא. אפשר למשל, לכתוב את תוכנית צבירת רבועי המספרים מ-1 עד 50 בצורה הבאה:

```
      MØNE=0
      DØ 9 K=1,50
      MØNE=MØNE+K**2
    9 CØNTINUE
```

<u>תרגיל</u>

נתונה טבלה באורך 100 איברים. יש למצוא את הערך הגבוה ביותר ולהעבירו למשתנה BIG. את מספר האיבר היחסי בטבלה יש לשמור במשתנה KEEP. התוכנית להתרת הבעיה:

<u>שורה</u>

```
1         DIMENSIØN  A(100)
2         BIG=A(1)
3         KEEP=1
4         DØ 888 I=2,100
5         IF (BIG.GE.A(I)) GØ TØ 888
6         BIG=A(I)
7         KEEP=I
8     888 CØNTINUE
9         STØP
10        END
```

בשורה 1 קובעים את שם המערך ואת גודלו. בשורות 2 ו-3 קובעים את תנאי ההתחלה של הבעיה: במשתנה BIG מציבים את הערך הראשון של המערך ובמשתנה KEEP, את המספר 1 בהתאמה.

שורות 4 עד 8 מהוות את הלולאה: ההוראה DØ בשורה 4 היא הוראת הלולאה וההוראה CØNTINUE בשורה 8 היא הוראת המשך המהווה ציון סוף הלולאה - מספרה 888, כפי שמוזכר גם בהוראת DØ. האינדקס של הלולאה משתנה מ-2 עד 100. התחלנו ב-2 כי אין צורך להשוות את הערך הראשון בטבלה לעצמו.

ההוראה המרכזית בשורה 5 משווה את הערך של המשתנה BIG אל הערך התורן במערך, בהתאם לערכו של אינדקס I ((A(I)). אם הערך של המשתנה BIG גדול או שווה לערך של המשתנה התורן במערך - אין כל שנוי ועוברים להוראה מספר 888(CØNTINUE) בשורה 8. זהו סוף הלולאה, ומכאן נחזור להוראה שלאחר ה-DØ אם האינדקס I טרם הגיע לערך הגבוה שלו - 100. אם התנאי בהוראה IF לא מתמלא - כלומר הערך של האיבר במערך גדול מ-BIG, מעבירים אותו למשתנה BIG ואת מספרו הסדורי למשתנה KEEP (שורות 7,6). עוברים להוראה CØNTINUE המחזירה לראש הלולאה.

עם סיום הלולאה עוברים להוראה STØP בשורה 9 ומסיימים.

קידום האינדקס בהוראת DØ

קידום האינדקס בהוראת DØ נעשה בדרך כלל בצעדים של יחידות, כלומר: 1, 2, 3 או 19,18,17 וכו'. לעתים יש צורך לקדם את האינדקס בדלוג שונה, למשל בזוגות.

לדוגמא: יש לחשב את סכום כל המספרים הזוגיים מ-22 עד 48 ולהדפיס את הסכום.

התוכנית:

```
   N=0
   DØ 5 I=22,48,2
 5 N=N+I
   PRINT 9, N
 9 FØRMAT (I4)
```

במקרה זה השתמשנו באופציה של הוראת DØ והיא ציון הפרמטר השלישי של האינדקס - פרמטר הדילוג. כלומר ערכי I יהיו 26,24,22 .... 48. בצורה דומה אפשר לקבוע לאינדקס הלולאה דילוגים אחרים.

קריאה וכתיבה של מערכים

כאשר יש צורך לקרוא סדרת נתונים לתוך טבלה או כאשר יש להדפיס נתונים מתוך טבלה - משתמשים בהוראות קריאה וכתיבה של מערכים.

הוראות קריאה וכתיבה יכולות להופיע בתוך לולאה ככל הוראת ביצוע אחרת:

דוגמא לקריאה והדפסה:

```
   K=15
   DØ 7 N=1,K
   READ 999, A
   B=A*5.3
 7 PRINT 888, B
```

לולאה זו תבוצע 15 פעם (לפי ערכו של K): קריאה של ערך A מכרטיס, כפל והדפסה של התוצאה.

שים לב שערך הגבול של האינדקס מוצג כמשתנה שערכו נקבע במהלך התוכנית, לפני הכניסה ללולאה.

דוגמא להדפסת נתונים מטבלה:

נתון וקטור A בן 20 איברים שיש להדפיס את ערכיהם. ההוראה PRINT תבוצע בלולאה 100 פעמים:

התוכנית:

```
      DØ 33 N=1,20
      PRINT 50, A(I)
   33 CØNTINUE
   50 FØRMAT (....)
```

כיון שפעולות קלט-פלט הקשורות במערכים הן תדירות למדי מאפשרת השפה צורות נוספות של פקודות קריאה - כתיבה המיוחדות לטיפול במערכים. הדבר מתבטא בהכנסת הלולאה לתוך ההוראה. הצורה שתתאים לקריאת הוקטור A בעל 20 האיברים תהיה:

```
      PRINT 50, (A(I), I=1,20)
```

ופירושה: הדפס לפי פורמט 50 את האיברים A(1), A(2), ..., A(20) של הוקטור A.

באופן כללי יהיו ההוראות לקריאה וכתיבה של וקטור חד-מימדי מהצורה:

$$\text{READ K, (V(i), i=}n_1,n_2\text{)}$$

$$\text{PRINT K, (V(i), i=}n_1,n_2\text{)}$$

כאשר K - הוא המספר של הוראת האיפיון, V - הוא שם הוקטור אותו אנו מעונינים לקרוא או לכתוב, n_1 ו-n_2 מציינים את תחום האיברים בוקטור V אותם יש לקרוא או לכתוב. איברים אלו יכולים להיות קבועים או משתנים שלימים.

אם בתוכנית מופיעות הפקודות

```
      M=2
      K=4
```

ואחר כך הפקודה

```
      READ 100, (LIST (I), I=M,K)
  100 FØRMAT (I5)
```

יתבטא הדבר בקריאת האלמנטים

```
      LIST(2), LIST(3), LIST(4)
```

אשר כל אחד מהם מנוקב בכרטיס נפרד, בחמישה הטורים הראשונים שבו.

תרגיל:

נתונה כרטסת המכילה ערכים, ערך אחד בכרטיס. בראש החבילה ישנו "כרטיס מדריך" שבו מנוקב מספר הכרטיסים (עם ערכים) בכרטסת והשנה שבה היא הוכנה.

פעולות העיבוד הנדרשות:

א. אם התאריך נמוך מ-1975 יש להדפיס הודעת שגיאה ולהפסיק את העבודה.
ב. יש לסכם את הערכים המנוקבים בכרטיסים ולחשב את הממוצע. את התוצאות יש להדפיס בסיום העבודה.

מבנה הקלט:

"כרטיס מדריך": ערך N מנוקב בטורים 1 - 2
תאריך (שנה) " 3 - 6
כרטיס נתונים : ערך X מנוקב בטורים 1 - 7
כמספר ממשי עם 2 ספרות עשרוניות.

מבנה הפלט:

הודעה על שגיאה בכרטיס מדריך:

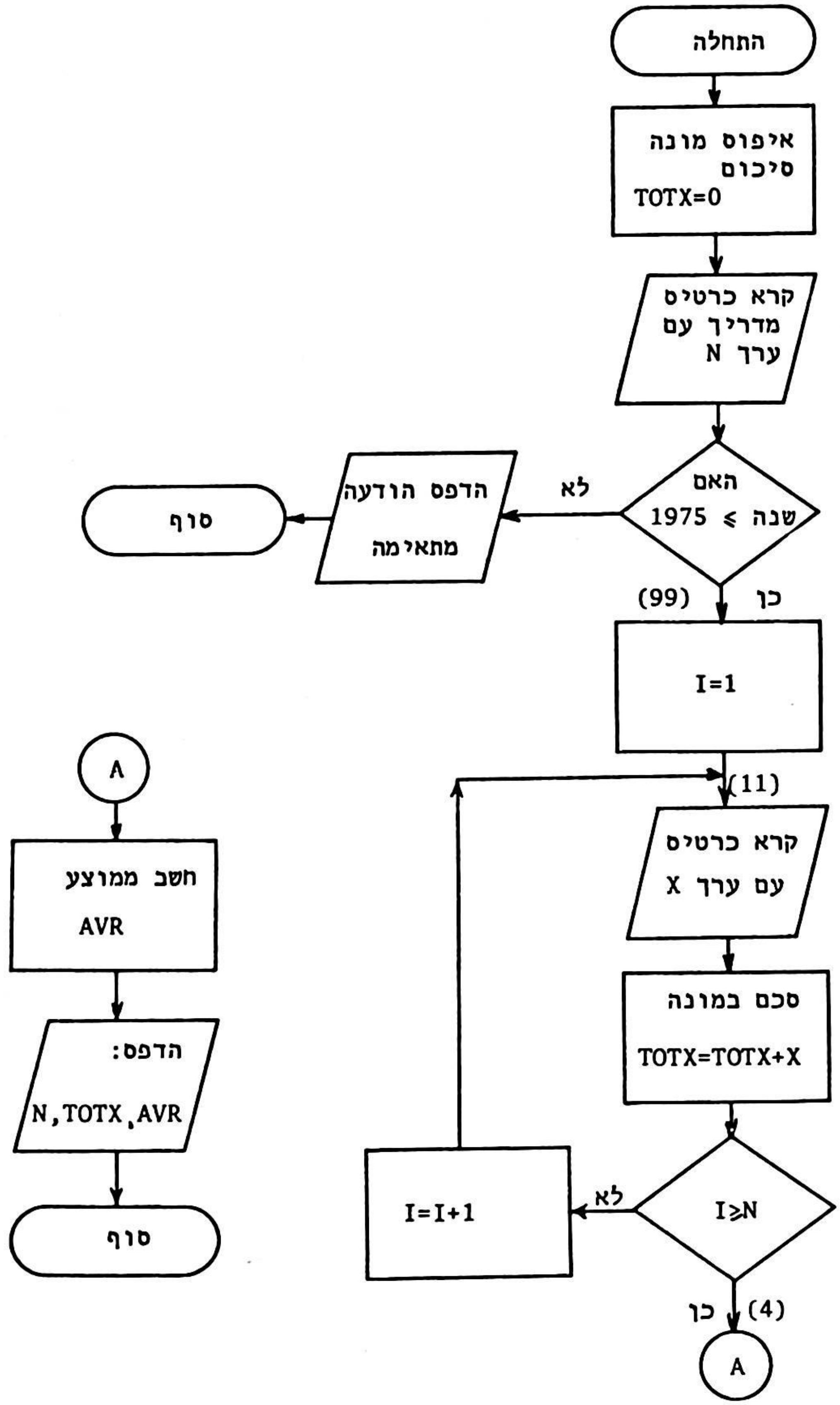
ERRØR.FILE DATE LESS THAN 1975: XXXX

תוצאת החישוב:

N=XX--TØTAL=XXXXXXXX.XX--AVERAGE=XXXXX.XX

2 רווחים 2 רווחים

<u>התרשים המלבני</u> (לבקרת לולאה ע"י הוראת IF)



FORTRAN Coding Form

תוכנית עם לולאת IF לבקרת הלולאה

```
      PROGRAM IFTEST (INPUT, OUTPUT)                                    IF  10
      TOTX=0.                                                           IF  20
      READ 1, N, IDATE                                                  IF  30
    1 FORMAT (I2,I4)                                                    IF  40
      IF (IDATE .GE. 1975) GO TO 99                                     IF  50
      PRINT 8, IDATE                                                    IF  60
    8 FORMAT (1H0,19HERROR IF FILE DATE:,I4)      (הכותרת מכילה 19 תוים) IF  70
      STOP                                                              IF  80
   99 I=1                                                               IF  90
   11 READ 2, X                                                         IF 100
    2 FORMAT (F7.2)                                                     IF 110
      TOTX = TOTX + X                                                   IF 120
      IF ( I .EQ. N ) GO TO 4                                           IF 130
      I = I+1                                                           IF 140
      GO TO 11                                                          IF 150
    4 AVR = TOTX / N                                                    IF 160
      PRINT 7, N, TOTX , AVR                                            IF 170
    7 FORMAT (1H1,2HN=, I2, 8H  TOTAL=,F11.2,10H  AVERAGE=,F8.2)        IF 180
      STOP                                                              IF 190
      END                                                               IF 200
```

תוכנית עם לולאת DO לבקרת הלולאה

```
      PROGRAM DOLOOP (INPUT, OUTPUT)                                    DO   1
                                                                        DO
      TOTX = 0.                                                         DO   2
      READ 1,N,IDATE                                                    DO   3
    1 FORMAT (I2,I4)                                                    DO   4
                                                                        DO
      IF (IDATE .GE. 1975) GO TO 99                                     DO   5
                                                                        DO
      PRINT 8,IDATE                                                     DO   6
    8 FORMAT (1H0,19HERROR IN FILE DATE:,I4)                            DO   7
      STOP                                                              DO   8
   99 DO 66 I=1,N,1                                                     DO   9
      READ 2,X                                                          DO  10
    2 FORMAT (F7.2)                                                     DO  11
   66 TOTX = TOTX + X                                                   DO  12
      AVR = TOTX / N                                                    DO  13
      PRINT 7,N,TOTX,AVR                                                DO  14
    7 FORMAT (1H1,2HN=,I2,8H  TOTAL=,F11.2,10H  AVERAGE=,F8.2)          DO  15
      STOP                                                              DO  16
      END                                                               DO
```

הסבר לתוכנית IFTEST

שורה 1 - שם התוכנית - IFTEST. הוראת הנחייה למהדיר.

2 - תנאי התחלה: איפוס מונה.

3 - 8 - קריאת כרטיס מדריך ובדיקת התאריך. אם התאריך שוה או גדול מ-1975 ממשיכים ואם התאריך נמוך מ-1975 מדפיסים הודעה ומפסיקים.

9 - תנאי התחלה לפני הלולאה המרכזית בתוכנית: הצבת האינדקס על ערכו הראשוני.

15-10 - הלולאה המרכזית בתוכנית: קריאת כרטיס, חבור הערך X למונה TØTX, בדיקה לסיום קריאת הכרטיסים. בכל בצוע של הלולאה משוים את האינ-דקס לערך N (מס' הכרטיסים לפי הנתון). אם האינדקס נמוך מ-N מוסי-פים 1 לאינדקס וחוזרים לראש הלולאה, ואם האינדקס שוה ל-N עוברים לקטע הסיום.

19-16 - קטע הסיום של התוכנית: חישוב הממוצע והדפסת שורה.

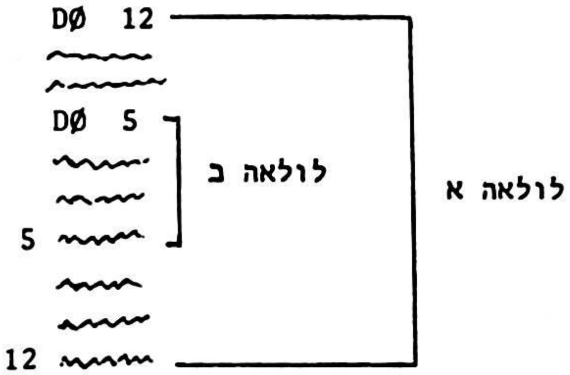
20 - הוראת הנחייה למהדיר שזהו סוף התוכנית.

הסבר לתוכנית DØLØØP

בתוכנית זו מבוקרת הלולאה המרכזית של התוכנית ע"י הוראת DØ (שורות 12-9). בתוכנית זו נחסכות 4 הוראות אשר שימשו בתוכנית הקודמת לבקרת הלולאה, ונוספה ההוראה DØ.

לולאות מוכלות

קיימת אפשרות להכיל לולאה בתוך לולאה. כלומר, בתוך קטע תוכנית המהווה לולאה אפשר להכיל לולאה המקיפה חלק כלשהו של ההוראות. אפשר לתאר זאת בצורה סכימטית בצורה הבאה:



נדגים עתה כיצד ניתן לקרוא ערכים לתוך מטריצה. נשתמש בדוגמא זו למטרה נוספת - להראות איך אפשר להפעיל לולאה בתוך לולאה.

<u>תרגיל:</u>

נתונה מטריצה בעלת 2 שורות ו-3 עמודות. הערכים מנוקבים בכרטיסים, בכל כרטיס - ערך אחד. יש לקרוא את הכרטיסים ולרשום את הערכים בתוך המטריצה.

```
            DIMENSIØN MATRIX(2,3)
  ┌─        DØ 9 I=1,2
  │   ┌─    DØ 8 J=1,3
א │ ב │     READ 100, MATRIX(I,J)
  │   └─ 8  CØNTINUE
  └─     9  CØNTINUE
```

לפנינו שתי לולאות: לולאה א היא "החיצונית" ולולאה ב היא "הפנימית". בפעם הראשונה כשתבוצע לולאה א יינתן ל-I הערך 1. וכאשר תבוצע הלולאה ב יינתן ל-J הערך 1. לאחר מכן תתבצע ההוראה READ לקריאת האיבר MATRIX(1,1). הוראת המוצא של הלולאה ב היא 8 CØNTINUE אשר תגרום להוספת 1 לאינדקס J ותחזיר את הפיקוח לראשית הלולאה ב, כלומר כעת J=2. הוראת הקריאה תתיחס כעת לאיבר MATRIX(1,2). האינדקס J יקודם שוב והוראת הקריאה תתיחס לאיבר MATRIX(1,3). כלומר, נקראו כל האיברים בשורה הראשונה במטריצה.

כאן מסתיימת פעולת לולאה ב והיציאה ממנה מובילה אל הוראה 9 CØNTINUE. הוראה זו גורמת לקידום האינדקס I ב-1, כלומר I=2 והיא מחזירה אותנו אל ראשית הלולאה, אל ההוראה DØ 8 J=1,3.

ושוב חוזר התהליך כאשר האינדקס I=2 והאינדקס J משתנה מ-1 ועד 3. תהליך זה יגרום קריאת האיברים של השורה השניה במטריצה: MATRIX(2,1); MATRIX(2,2); MATRIX(2,3). עם סיום הלולאה I שוב נעבור אל הוראת CØNTINUE 9 אשר תקדם את I ל-3 ותפעיל את הלולאה I כאשר J משתנה מ-1 עד 3 לקריאת האיברים של השורה השלישית במטריצה.

נקודות המוצא של הלולאות א ו-ב הן סמוכות, ומכיון שאין שוני בין ההו-ראות בנקודות המוצא של שתיהן, אפשר ללכד אותן בצורה הבאה:

```
              DØ 4 I=1,2
 א            DØ 4 J=1,3
       ב      READ 100, MATRIX(I,J)
            4 CØNTINUE
```

אפשר, במקרה זה, לרשום את התוית 4 ליד ההוראה READ ללא צורך בהוראת CØNTINUE:

```
      DØ 4 I=1,2
      DØ 4 J=1,3
    4 READ 100,MATRIX(I,J)
```

את הוראות בצוע הלולאה לקריאת המטריצה אפשר לכלול, כפי שראינו קודם, בהוראת ה-READ (או בהוראת WRITE או PRINT כאשר מדובר בפלט).
בצורה זו:

```
      DØ 5 I=1,2
    5 READ 100,(MATRIX(I,J),J=1,3)
  100 FØRMAT (3F10.1)
```

או בצורה זו: `READ 100,((MATRIX(I,J),J=1,3),I=1,2)`

בצוע הפקודות מתבטא בקריאת איברי המטריצה לפי סדר השורות: I קבוע בערך ראשון וערכי J משתנים בכל התחום שלהם, I מתקדם לערך הבא ושוב משתנים ערכי J וכן הלאה, עד גבול התחום של I. אפשר לכתוב את הערכים לפי הסדר הבא:

MATRIX(1,1), MATRIX(1,2), MATRIX(1,3), MATRIX(2,1), MATRIX(2,2), MATRIX(2,3), MATRIX(3,1), MATRIX(3,2), MATRIX(3,3).

בדוגמאות הקודמות הנחנו שבכל כרטיס מנוקבים 3 ערכים, כלומר - שורה אחת במטריצה. כדי לקרוא את כל המטריצה יש לקרוא 6 כרטיסים (2×3). אם מנוקב בכר-טיס ערך אחד בלבד וכך יוגדר גם בהוראת האיפיון, לא תשתנינה ההוראות לקריאת המטריצה. הוראת READ תחזור על עצמה 3 פעמים כדי להשלים קריאת 3 ערכים עבור שורה אחת. כלומר, הוראת האיפיון קובעת כמה פעולות READ תבוצענה למעשה כדי להשלים את רשימת המשתנים אשר מופיעים בהוראת ה-READ (וכן בבצוע הוראות פלט).

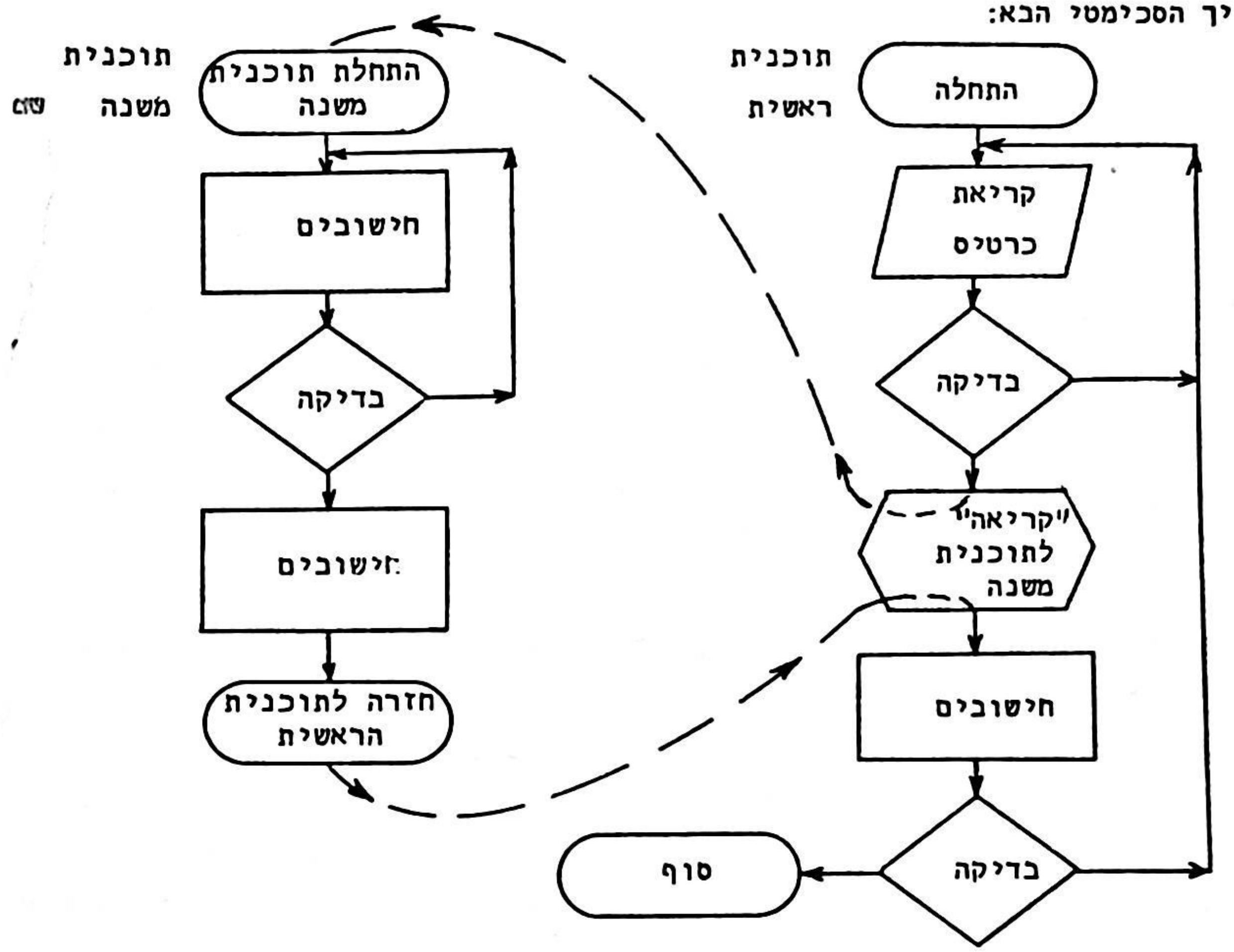
פרק 4: תוכניות משנה- Subprograms

מבנה תוכנית

תוכנית עיבוד מורכבת מקטעים שונים המבצעים משימות שונות בתהליך העיבוד. קטעים מסוימים בתוכנית יכולים לחזור על עצמם, כמו למשל חישוב שורש רבועי, הדפסת כותרת דף חדש, חישוב פונקציה טריגונומטרית מסוימת, מציאת הערך הגדול מתוך ערכים נתונים ועוד. גם אם קטע מסוים לא חוזר על עצמו באותה תוכנית, נמצא שהוא חוזר בתוכניות שונות. כדאי אם כן, לשמור את אותם קטעי תוכניות ולנצלם שוב ושוב ללא צורך בכתיבה מחדש. קטעים אלו הן תוכניות משנה - subprograms.

תוכניות המשנה נשמרות בספרית תוכניות (program library) והתוכנית הראשית, "קוראת" להן. תפקידו של המתכנת לדאוג לשילוב הלוגי שלהן בתוכניתו והמהדיר (Compiler) מטפל בשילוב הפיזי שלהן. ספרית תוכניות המשנה נתונה לשימושו של כל אחד והוא פטור על כן מלכתוב את אותן התוכניות מחדש, אם הן מתאימות לצרכיו.

נתבונן בנוהל השילוב של תוכניות המשנה ובשיטה בה מבוצע העיבוד בעזרת התהליך הסכימטי הבא:



בשעת הפעלת התוכנית הראשית מבוצעות ההוראות השונות בזו אחר זו, עד אשר מגיעים להוראת "קריאה" לתוכנית משנה. בנקודה זו, מבוצע מעבר להוראה הראשונה בתוכנית המשנה וממנה ממשיכים בבצוע ההוראות בזו אחר זו עד אשר מסַימים אותה. כאן חוזרים אל התוכנית הראשית, אל ההוראה העוקבת את הוראת הקריאה לתוכנית המש-נה. מכאן ממשיך תהליך העיבוד בתוכנית הראשית עד לסיומה או עד אשר פוגשים שוב בהוראת קריאה לתוכנית המשנה הנוכחית או לתוכנית משנה חדשה.

קיימים סוגים אחדים של תוכניות משנה, נדון כאן בכמה מהן.

פונקציה בנויה או פונקצית ספריה

כל העוסק בתכנות מדעי נתקל לעיתים קרובות במספר פונקציות כגון הפונקציות הטריגונומטריות, לוגריתמים, שורש ריבועי וכיו"ב. כדי להקל על המתכנת מצורפת לפורטרן ספריה של תוכניות אשר מחשבות פונקציות כאלה ואשר אפשר לשלבן בצורה פשוטה לתוך תוכנית הפורטרן. הפניה לתוכניות אלה, הקרויות "פונקציות-בנויות", מתבצעת על-ידי ציון שמה של התוכנית והפרמטרים הנדרשים לצורך החישוב.

התוכנית לחישוב השורש הריבועי, למשל, נקראת SQRT. אם קיים בתוכנית ֹתנה בשם X ומעונינים לחשב את $Y=\sqrt{X}$ תיכתב ההוראה באופן הבא:

```
Y=SQRT(X)
```

לשם חישוב הפונקציה סינוס נכתוב:

```
T=SIN(P)
```

כלומר, כתיב המזכיר שוב את הכתיב המתימטי.

המשתנה של פונקציה-בנויה יכול להיות קָבוע, משתנה, או ביטוי אריתמט כמו-כן מותר לקרוא לפונקציה של פונקציה. למשל:

```
Z=SIN(SQRT(X**2+Y**2))
```

פונקציה בנויה יכולה להופיע גם כחלק של ביטוי אריתמטי:

```
Y=3.0*SIN(X)+SQRT(X**2+CØS(W))
```

סיפרית הפונקציות הבנויות שונה בניבים השונים של השפה, אם כי רובן קיימות בכל הניבים.

פונקציות נוספות נפוצות הן:

EXP(X) - לחישוב הערך e בחזקת X;

ABS(X) - למציאת הערך האבסולוטי של X;

ALØG10(X) - למציאת הלוגריתם העשרוני של X;

ועוד.

הפונקציות האלו מטפלות בערך ממשי בודד (Real number) והתוצאה גם היא ערך ממשי.

יש פונקציות הפועלות גם על ערכים שלימים (Integers), ואז מוסיפים את האות I משמאל לשם הפונקציה, למשל: M=IABS(K).

<u>כללי כתיבה של פונקציות:</u>

כל מתכנת יכול לכתוב בעצמו פונקציות ולהוסיפן לספרית התוכניות. נציין, כאן מספר כללים אשר יאפשרו לנו לכתוב פונקציות פשוטות:

1. פונקציה יכולה לטפל במשתנה אחד או יותר (פרמטרים), אולם תמיד היא מחזירה ערך אחד בלבד.
2. ההוראה הראשונה בתוכנית הפונקציה מציינת את שמה ולאחריו, בסוגריים, רשי-מת הפרמטרים. הערך הסופי מתקבל למשתנה ששמו כשם הפונקציה.
3. הקריאה לפונקציה היא לפי שמה כשלאחריו נרשמים הפרמטרים בתוך סוגריים.
4. הפרמטרים הנמסרים לפונקציה צריכים להתאים בסוגם (משתנים ממשים או שלימים), מספרם ובסדר הופעתם לפרמטרים המופיעים בהגדרת הפונקציה. שמות המשתנים הנמסרים לפונקציה אינם בהכרח זהים לאלו המוגדרים בה ולכן <u>חשובים</u> הסוג, המספר והסדר.
5. חזרה מהפונקציה אל התוכנית הראשית היא ע"י הוראת RETURN.

סוף התוכנית המהווה את הפונקציה נקבע ע"י הוראת END.

נשתמש בכללים אלו לשם כתיבת פונקציה:

נתונה כרטסת של 150 תלמידים. בכל כרטיס מנוקבים מס' התלמיד, שם התלמיד ו-3 ציונים. יש לחשב את ממוצע הציונים ולהדפיסו.

כתיבת התוכנית:

התוכנית הראשית תטפל בקריאה והדפסה של הנתונים.

תוכנית-המשנה תבצע חישוב ממוצע של 3 ערכים ותמסור את התוצאה לתוכנית הראשית.

<u>הפונקציה:</u>

```
FUNCTIØN AVR (K1,K2,K3)
M=K1+K2+K3
AVR=M/3
RETURN
END
```

הסבר:

שם הפונקציה הוא AVR (מספר ממשי) והיא מקבלת 3 פרמטרים של 3 מספרים שלימים.
תוצאת החישוב של הפונקציה מקנה את התוצאה למשתנה ששמו כשם הפונקציה.

<u>התוכנית הראשית:</u>

```
   PRØGRAM .....
   DØ 9  I=1,150
   READ 5, M, J, L
   TØTAL=AVR(M,J,L)+5.0
9  PRINT 8, TØTAL
   STØP
5  FØRMAT (3I2)
8  FØRMAT (7H RESULT=,F4.1)
   END
```

הסבר:

התוכנית מבצעת בלולאה, 150 פעם, קריאה של כרטיסים לפי איפיון מס' 5, חישוב מסוים והדפסת תוצאת החישוב לפי איפיון מס' 8.
בתהליך החישוב מופעלת הפונקציה AVR אשר הוכנה על ידנו קודם לכן. הפרמטרים הנמסרים לה הם שלשה ערכים שלימים M, J, L שנקראו מכרטיסים. תוצאת החישוב של הפונקציה היא הערך הממשי הבודד AVR המהווה את הממוצע שלהם. אל ערך זה נחבר את המספר 5.0 והתוצאה TØTAL תודפס בהוראת PRINT העוקבת.

פונקצית הוראה

המתכנת יכול לכתוב לעצמו פונקציות פשוטות שניתן להכילן בביטוי אריתמטי אחד. פונקציות אלו נכתבות בראש התוכנית בה הן מיועדות להפעלה ואינן דורשות כל הכנה מיוחדת או תווך של ספרית התוכניות.

לדוגמא:

נשתמש בפונקצית-הוראה לשם חישוב של סכום שורשים רבועיים של שני ערכים נתונים.

מבנה התוכנית:

```
      PRØGRAM STMT(INPUT,ØUTPUT)
      SUM(X,Y)=SQRT(X)+SQRT(Y)
      READ 5, A, B
      IF (A.EQ.B) GØ TØ 6
      C=SUM(A,B)*7.5
    6 D=A+B
      .....
      END
```

הסבר:

פונקציה SUM מקבלת שני ערכים ממשיים ומחשבת את סכום השורשים שלהם. בתוכנית היא מקבלת למעשה את הערכים A ו-B אשר נתקבלו ע"י הוראת READ. הערך היחיד שמוסרת הפונקציה נכנס לתהליך החישוב של המשתנה C.

הכללים העיקריים לכתיבת הפונקציה הם:

1. הגדרת הפונקציה היא ע"י ביטוי יחיד אשר בו מופיע שם הפונקציה ולאחריו בסוגריים - רשימת הפרמטרים. מימין לסימן השויון מופיע ביטוי אריתמטי.
2. פונים אל הפונקציה ע"י הכרזת שמה כשלאחריו בסוגריים רשימת הפרמטרים המתאימים בסוג ובסדר לאלו המופיעים בהגדרת הפונקציה. הפונקציה מחזירה ערך יחיד שסוגו כשם הפונקציה (משתנה ממשי או שלם).
3. אפשר להשתמש בפונקציה בתוך פונקציה.
4. הגדרת הפונקציה טובה רק בתוכנית בו היא נכתבה.
5. את הגדרת הפונקציה רושמים לפני הוראות הבצוע של התוכנית.

שגרה (subroutine)

שגרה היא תוכנית משנה אשר מיועדת לבצע סדרה מוגדרת של פעולות. אפשר לפנות אל שגרה מתוך התוכנית הראשית או מתוך שגרה אחרת. בכל פניה אל שגרה יש לציין בסוגריים את רשימת המשתנים המועברים אליה כנתונים ואת רשימת המשתנים שמקבלים ממנה עם תוצאות. שמות משתנים אלו מוכרים בתוכנית הראשית. נראה אחר כך שכאשר מכינים שגרה צריכים לציין בהוראת הכותרת שלה גם כן רשימת משתנים, אשר מקבילה לרשימה אשר נמסרת לה בשעה שפונים אליה מתוך התוכנית הראשית.

נדגים את השימוש בשגרה בעזרת דוגמא פשוטה:
יש לקרוא 5 כרטיסים שבהם מנוקבים ערכי אורך ורוחב של מלבנים. יש לחשב, בעזרת שגרה, את השטח ואת ההיקף של כל מלבן ולהדפיס את התוצאה.

<u>התוכנית הראשית:</u>

```
      PRØGRAM CALC (INPUT,ØUTPUT)
      DØ 1 I=1,5
      READ 7,A,B
      CALL FRAME(A,B,AREA,PRMTR)        סיעוף לשגרה
    1 PRINT 10,AREA,PRMTR
      STØP
    7 FØRMAT (2F3.1)
   10 FØRMAT (7H0bAREA=,F5.1,6HbPRMTR=,F4.1)
      END
```

<u>השגרה:</u>

```
      SUBRØUTINE FRAME(X,Y,AREA,LNGTH)
      AREA=X*Y
      PR=2*(X+Y)
      RETURN
      END
```

הסבר:
התוכנית הראשית נקראת CALC. היא קוראת מכרטיסים את ערכי האורך והרוחב של המלבנים שצריך לחשב. בכל כרטיס ישנם שני שדות A, B אשר קוראים אותם לפי

הוראת איפיון מס' 7. כל התוכנית הראשית מבוקרת ע"י הוראת הלולאה אשר מב-צעת את גוף התוכנית חמש פעמים, פעם אחת עבור כל כרטיס, החל מההוראה העוקבת את הוראת DØ וכולל ההוראה שתוית שלה היא 1 (ההוראה PRINT).

כאשר צריך לבצע את החישוב מתבצע סיעוף אל השגרה אשר מיועדת למטרה זו. הסיעוף או הפניה אל השגרה נעשה בעזרת ההוראה CALL כאשר לאחריה נרשם שם השגרה (FRAME) ולאחריו, בסוגריים, רשימת הפרמטרים. במקרה זה כוללת רשימת הפרמטרים ארבעה משתנים A, B, AREA, PRMTR. שני המשתנים הראשונים ברשימה נושאים את נתוני הבעיה ושני המשתנים האחרונים ברשימה יחזירו מהשגרה את תוצאות החישוב.

נניח שהשגרה בוצעה ועכשיו, כאשר חוזרים ממנה יש לנו שני משתנים (האחרונים ברשימה) שבהם יש ערכים לפי תנאי הבעיה. עתה עלינו להדפיס את התוצאות לפי הו-ראת איפיון 10. בהוראה זו הקדמנו לפני כל מבנה את ההסבר השייך לו, בעזרת האי-פיון H (שים לב למספר שמשמאל לאות H המציין את מספר התוים שמימין לאות H). הדפסת התוצאה תופיע ברווח של שתי שורות מהשורה הקודמת (מכיון שהתו הראשון בשורה - תו הקוד - הוא 0), ותוכנה יהיה כזה:

ƀAREA=XXX.XƀPRMTR=XX.X

לאחר השלמת בצוע הלולאה (לפי ההוראה DØ) 5 פעמים, עוברים להוראה העוקבת את הוראת PRINT. הוראה זו היא STØP וכאן מסתיימת התוכנית. בתוכנית כלולות שתי הוראות האיפיון (FØRMAT) אשר אפשר לרשום אותן בכל מקום, לפני ההוראה END מציינת למהדיר את הכרטיס האחרון בתוכנית.

נעבור עתה אל השגרה:

ההוראה הראשונה בשגרה היא הוראת הכרזה SUBRØUTINE כשלאחריה מופיע שם השגרה (FRAME), ולאחריו בסוגריים רשימת פרמטרים. רשימת הפרמטרים שהופיעה בהוראת CALL הפונה לשגרה מקבילה לרשימה זו. שמות הפרמטרים אינם חייבים להיות זהים אולם סוגם, מספרם והסדר שלהם חייבים להיות זהים.

השגרה מניחה שנתוני הקלט מתקבלים בפרמטרים X, Y והיא משתמשת בהם לחישוב התוצאות AREA, LNGTH המייצגים את השטח וההיקף.

החזרה מהשגרה היא ע"י ההוראה RETURN. בסוף השגרה, ככל תוכנית מופיעה ההוראה END המיועדת עבור המהדיר.

דוגמא נוספת:

יש לעבד נתונים סטטיסטיים ולהפיק דוח. הנתונים מתקבלים בכרטיסים, 8 שדות בכרטיס, אשר כמה מהם יכולים להיות אפסים.

יש למצוא בכל סדרה (כרטיס) את הערך הקטן, הערך הגבוה ואת הממוצע. כמו כן יש לציין כמה ערכים שאינם אפס קיימים בכל סדרה. את התוצאות יש להדפיס עבור כל כרטיס.

התוכנית הראשית תטפל בקריאה ובהדפסה ובפעולות אחרות הנדרשות ממנה. השגרה תבצע את החישובים והבדיקות לפי תנאי הבעיה.

התוכנית הראשית:

```
PRØGRAM ....
DIMENSIØN A(8)
---------
READ 7,(A(I),I=1,8)
---------
CALL STAT(A,SMALL,BIG,N)
---------
PRINT 15,SMALL,BIG,N
---------
STØP
END
```

השגרה:

```
SUBRØUTINE STAT(TABLE,SM,BG,I)
DIMENSIØN TABLE(8)
---------
---------
RETURN
END
```

הסבר:

התוכנית הראשית קוראת את הנתונים מכל כרטיס אל וקטור בשם A בעל 8 איברים.
כדי שהשגרה תוכל לבצע את החישוב על 8 ערכים אלו יש להעביר אליה את הידיעה על וקטור זה בצורת פרמטר. השגרה עצמה צריכה לדעת לטפל בוקטור ועל כן צריכה להיות

בה גם כן הכרזה DIMENSIØN, כפי שקיימת בתוכנית הראשית. שם הוקטור יכול להיות שונה (כך יהיה המצב בדרך כלל) אולם מספר איבריו צריך להיות שוה. קיימת דרך להעביר לשגרה את מספר איברי הוקטור אשר ידוע בתוכנית הראשית, אולם לא נטפל בזאת כאן.

בהוראת CALL בתוכנית הראשית יש פניה לשגרה STAT עם פרוט של 4 פרמטרים: הראשון - פרמטר קלט - שם של הוקטור המכיל את הנתונים והשלשה הבאים - פרמטרים לפלט. במשתנים אלו תועבר התוצאה מהשגרה לתוכנית הראשית. שים לב שהפרמטר האחרון N מציין מספר שלם, בעוד השאר מציינים מספרים ממשיים.

בהוראת SUBRØUTINE של השגרה רשום שמה - STAT ולאחריו רשימה של 4 פרמטרים המקבילים בתכונותיהם וסדרם לאלו שבהוראת CALL. לאחר הוראת SUBRØUTINE רשומה הוראת DIMENSIØN המכריזה על ממדי הוקטור TABLE.

<u>כללי כתיבה של שגרות</u>

1. כפי שראינו, בכל שגרה יש הוראת הכרזה SUBRØUTINE, כשלאחריה שם השגרה ורשימת פרמטרים. הפרמטרים הם משתנים שלימים וממשיים. רשימת פרמטרים זו הינה הדרך (לא היחידה!) להעברת נתונים בין התוכנית הראשית אל השגרה ולהיפך.

2. קוראים לשגרה בעזרת ההוראה CALL, כשלאחריה מופיע שם השגרה ורשימת פרמטרים המתאימים לדרישות התוכנית הראשית שממנה פונים לשגרה. הפרמטרים יכולים להיות משתנים או מספרים קבועים.
   הסוג, הסדר ומספר הפרמטרים חייב להיות זהה לרשימה שבהוראת SUBRØUTINE, אולם אין חשיבות לשמות המשתנים.

3. הכניסה לשגרה מתוך התוכנית הראשית היא בהוראת CALL. החזרה מהשגרה היא בהוראת RETURN שלה הממשיכה את בצוע הפקודות שלאחר הוראת CALL. מותרות מספר הוראות RETURN בשגרה כשם שמותר כמה הוראות STØP בתוכנית ראשית.

   הערה: ישנן שגרות עם יותר מכניסה אחת. במקרה שלפנינו הנחנו שההוראה הראשונה בשגרה היא זו שלאחר ההוראה SUBRØUTINE.

פרק 5: תרגיל סיכום

תאור הבעיה

בבחירות לנשיאות המועדון עמדו לבחירה 5 מועמדים אופן ההצבעה של כל בוחר נוקב על כרטיס קלט באופן הבא.

טורים 1 - 7 מס' ת.ז. של המצביע
8 - 20 שם המצביע
25 המספר הסדורי של המועמד לנשיאות (1 עד 5)

את כרטיסי הבוחרים ריכזו לחבילה אחת ובסופה הוסיפו כרטיס ריק.
כתוב תוכנית שתבצע את הפעולות הבאות:

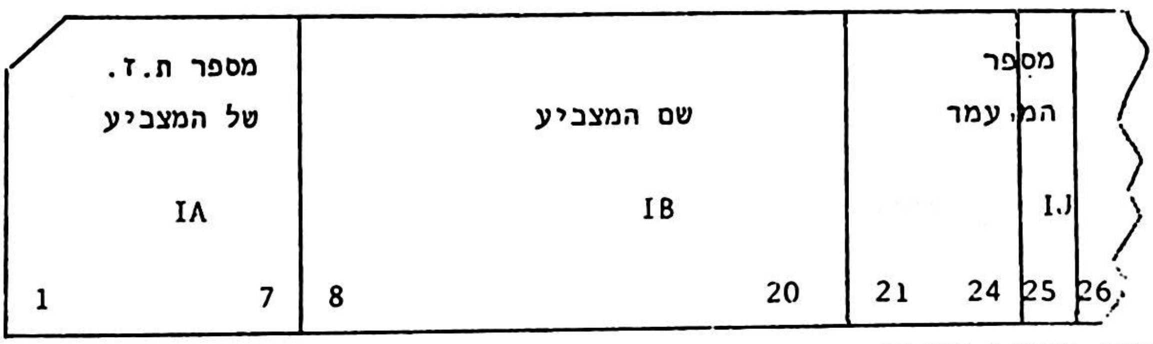
1. תדפיס את נתוני הקלט,
2. תקבע כמה אנשים השתתפו בבחירות ותדפיס את התוצאה,
3. תקבע ותדפיס מי מבין המועמרים זכה וכמה קולות קבל,
4. תחשב את יחס מספרי הקולות של הזוכה לשאר המועמדים ותדפיס את התוצאה.

יש לתת לדו"ח כותרת בראש דף חדש, ולאחר מכן להדפיס את גוף הדו"ח. יש לתת הסבר לנתונים שיודפסו, לדוגמא:

votes for winner XXXX

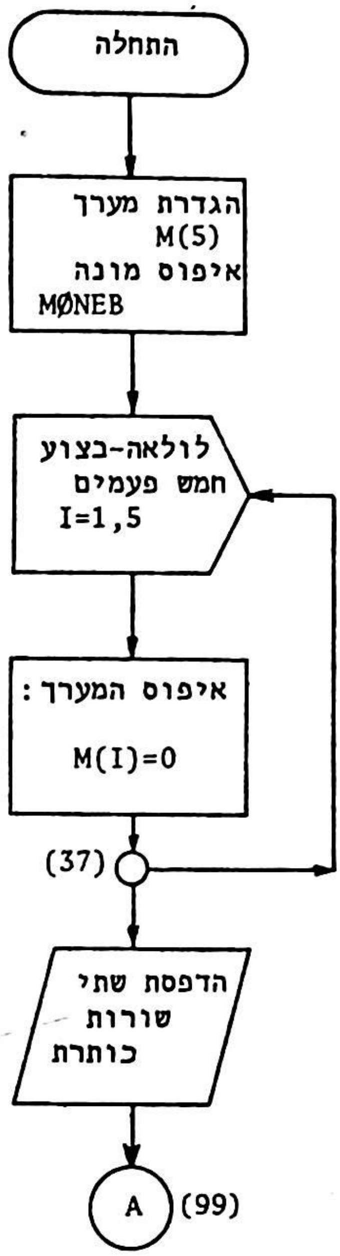
<u>התרת הבעיה:</u>

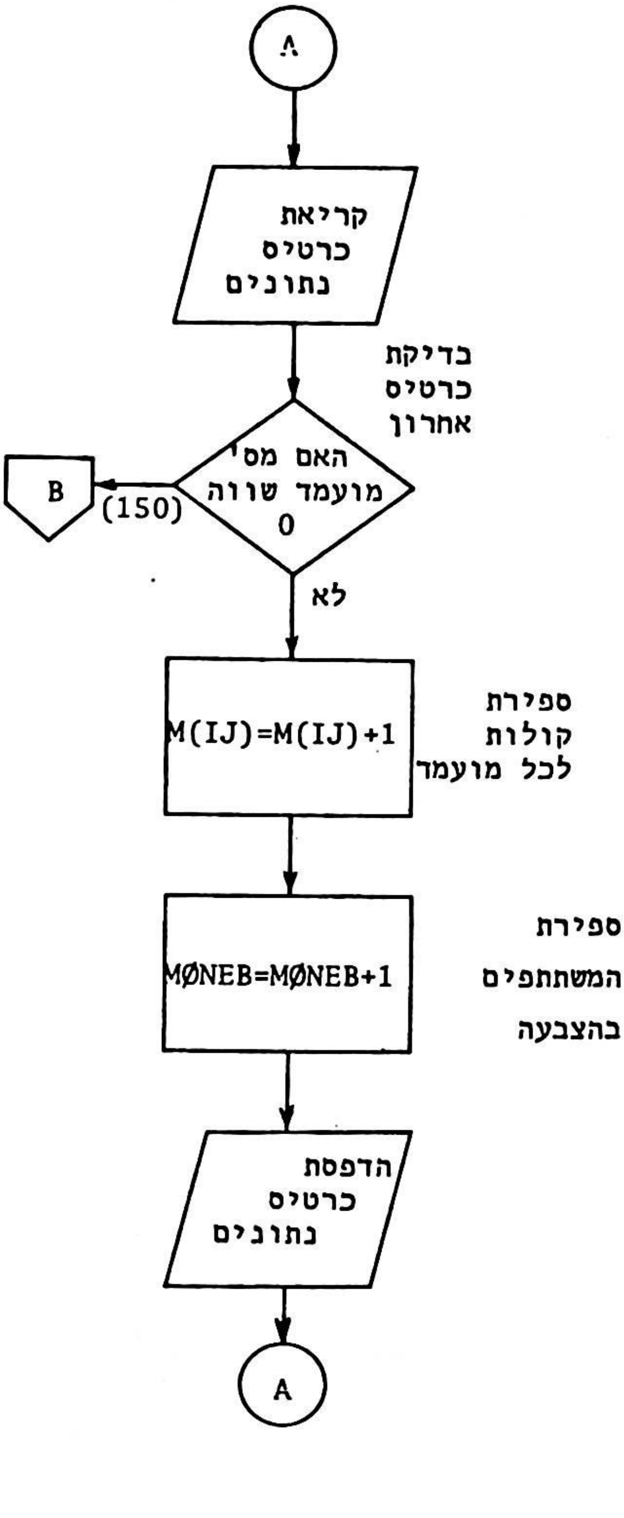
1. מבנה כרטיסי הקלט, עם שמות השדות:

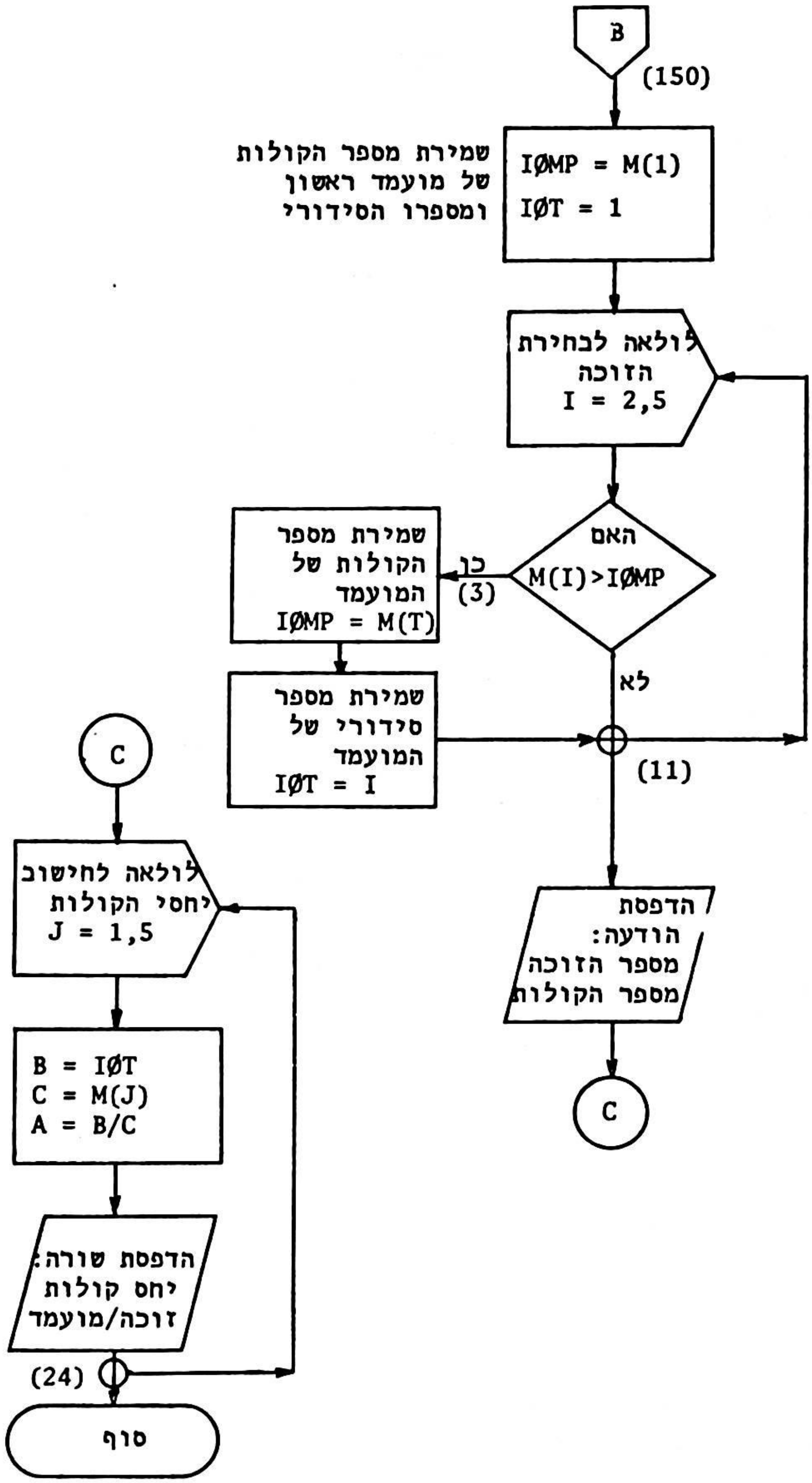


סימון כרטיס אחרון:
הכרטיס האחרון בקלט ישא את הסימון 0 (אפס) בטור 25 (שדה IJ)

<u>תרשים מלבני</u>






B
(150)
שמירת מספר הקולות
של מועמד ראשון
ומספרו הסידורי
IØMP = M(1)
IØT = 1
לולאה לבחירת
הזוכה
I = 2,5
האם
M(I)>IØMP
כן
(3)
שמירת מספר
הקולות של
המועמד
IØMP = M(T)
לא
שמירת מספר
סידורי של
המועמד
IØT = I
(11)
הדפסת
הודעה:
מספר הזוכה
מספר הקולות
C
C
לולאה לחישוב
יחסי הקולות
J = 1,5
B = IØT
C = M(J)
A = B/C
הדפסת שורה:
יחס קולות
זוכה/מועמד
(24)
סוף

<u>תכנות הבעיה</u>

```
                  PROGRAM ALEX(INPUT,OUTPUT)
000003            DIMENSION M(5)
000003            MONEB=0
000004            DO 37 I=1,5
000006         37 M(I)=0
000011            PRINT 38
000015         38 FORMAT(1H1,4X,*NO,     IDENT,            NAME          KOD*)
000015            PRINT 39
000021         39 FORMAT(1H ,3X,*------ *,9(*-*),5X,8(*-*),3X,5(*-*))
000021         99 READ 10,IA,IB,IJ
000033         10 FORMAT(I7,A10,7X,I1)
000033            IF(IJ.EQ.0) GO TO 150
000034            M(IJ)=M(IJ)+1
000036            MONEB=MONEB+1
000040            PRINT 100,MONEB,IA,IB,IJ
000053        100 FORMAT(1H0,4X,I3,3X,I7,3X,A10,6X,I1)
000053            GO TO 99
000054        150 IOMP=M(1)
000056            DO 11 I=2,5
000057            IF(IOMP.LE.M(I)) GO TO 3
000062            GO TO 11
000062          3 IOMP=M(I)
000064            IOT=I
000065         11 CONTINUE
000067            PRINT 12,IOT,IOMP
000077         12 FORMAT(1H0,*WINNER NO=*,I1,5X,*ACCEPT*,I3,* VOTES*)
000077            PRINT 13
000103         13 FORMAT(1H ,36(*=*))
000103            DO 24 J=1,5
000105            B=M(IOT)
000107            C=M(J)
000110            A=B/C
000113            PRINT 23,IOT,J,A
000124         23 FORMAT(1H ,*VOTE NO*,I1,* / *,*VOTE NO*,I1,*=*,F5.3)
000124         24 CONTINUE
000126            STOP
000130            END
007700
```

<u>הסבר לתוכנית</u>

שורה 1- התוכנית הופעלה במחשב CDC שבו יש צורך להקדים את תוכנית הפורטרן בהוראה PROGRAM. שם התוכנית ALEX.

2- הכרזה על מערך חד-ממדי M בן 5 איברים. במערך זה נצבור את מספר המצ-ביעים עבור כל מועמד. האיבר הראשון מיועד למועמד הראשון, האיבר השני - למועמד השני וכן הלאה.

3- איפוס המונה לספירת מספר המשתתפים בהצבעה.

5-4 הוראת לולאה. לולאה זו מיועדת לאיפוס איברי המערך כדי שאפשר יהיה לצבור בהם את מספר המצביעים לכל מועמד.

9-6 הדפסה של שתי שורות כותרת להדפסת נתוני הקלט: אחת טכסט ואחת קוי הדגשה. הכותרת הראשונה תודפס בראש הדף לפי קוד קידום הנייר = 1 (1H1). השורה השניה תודפס בצמוד לה לפי קוד קידום הנייר blank (1H_). שים לב לרווחים שבין חלקי הכותרת. יש לשמור על רווחים אלו כאשר כותבים את הוראת האיפיון להדפסת שורות הנתונים (שורה 16).

11-10 קריאה של כרטיס נתונים. קוראים שלשה שדות לפי תנאי הבעיה.

12- בדיקה לכרטיס אחרון, אם המספר בשדה IJ (טור 25 בכרטיס) שוה ל-אפס. אם התנאי מתקיים מבצעים סיעוף להוראה מס' 150 בשורה 18. אם התנאי אינו מתקיים - ממשיכים בהוראה העוקבת.

13- ספירת מספר המצביעים עבור כל מועמד: מוסיפים 1 לאיבר המתאים במערך M.

14- ספירת מספר המצביעים: מוסיפים 1 למונה.

16-15 הדפסת שורת נתונים. מוסיפים גם את השדה MONEB בצד שמאל כדי לציין מספר סדורי (אשר אינו מופיע בכרטיס). הוראת האיפיון צריכה להתאים לאיפיון הדפסת הכותרת (בשורה 7).

17- חזרה לקריאת הכרטיס הבא.

18- להוראה זו מגיעים כאשר מסתיימת קריאת נתוני הקלט (לפי הבדיקה בשורה 12). מכאן ואילך מבצעים פעולות המתבססות על נתונים וערכים שנמצאים בזכרון המחשב.
הפעולה הראשונה הנדרשת היא לבחור את המועמד הזוכה:
מעבירים למשתנה עזר זמני (IOMP) את מספר המצביעים עבור המועמד הרא-שון. כאן <u>דרושה</u> הוראה נוספת, אשר חסרה <u>בטעות</u>. נעמוד על כך בהמשך ונסביר את משמעות השגיאה.

24-19 לולאה להשואת מספר המצביעים עבור המועמדים לבחירת המועמד הזוכה: אם המספר שבאיבר המערך קטן או שוה מהמספר שבמשתנה העזר IOMP אין

משנים אותו. אך אם המספר שבמערך גדול מהמספר במשתנה - מעבירים אותו למשתנה IOMP ואת מספרו הסידורי למשתנה IOT (שורות 22-23).
כאן גילינו את השגיאה: צריך היה לרשום את ההוראה IOT=1 לאחר ההוראה IOMP=M(1) שבשורה 18. במצב הקיים, אם הזוכה הוא המועמד הראשון המש-תנה IOT איננו מכיל את המספר 1 כנדרש אך אם המועמד הזוכה איננו הראשון המשתנה IOT מקבל את המספר הסידורי הדרוש.

25-28 הדפסת הודעה על הזוכה: שורת טכסט ושורת הדגשה.

29-35 בקטע זה מבוצעת לולאה לחישוב והדפסת יחסי הקולות של המועמד הזוכה לשאר המועמדים. מעבירים את הערכים ממשתנים שלימים (Integers) למש-תנים ממשיים (Real) לבצוע החישוב. בלולאה זו חישבנו גם את יחס מספר הקולות של הזוכה לעצמו, לשם הפשטות של התכנות.

36- הוראה STOP עוצרת את העבודה בתוכנית זו ומעבירה את הפיקוד למערכת ההפעלה אשר תטען תוכנית אחרת ותפעיל אותה.

37- הוראה END מציינת את סיום הוראות התוכנית (הוראה זו איננה הוראת בצוע אלא מיועדת למהדיר).

תדפיס הפלט

NO.	IDENT.	NAME	KOD
1	1111111	DAVID EL	1
2	3020405	MOSHE IZ	2
3	3271573	ELIMELACH	3
4	2813172	KOTY	1
5	8913217	ZRAKOVSKY	5
6	8319071	DAN COHE	2
7	1234567	JAKOV MI	2
8	8910112	BEN LOLO	2
9	3181917	ALBERT	1
10	3215781	A.D. GORD	1
11	2468107	FRANKEL	2
12	3178132	BIALIK	5
13	1134001	DOV HOZ	2
14	3201702	MICAEL	4
15	3410210	ELCATRAZ	2
16	101010	ABG BEN	3
17	3217121	MEN POWER	2
18	5183917	BEN HORE	1
19	3781789	BEN YONA	4
20	2932475	GOLYAT	2
21	2389792	RABENUE	1

```
WINNER NO=2      ACCEPT  9 VOTES
-------------------------------------
VOTE NO2 / VOTE NO1=1,500
VOTE NO2 / VOTE NO2=1,000
VOTE NO2 / VOTE NO3=4,500
VOTE NO2 / VOTE NO4=4,500
VOTE NO2 / VOTE NO5=4,500
```

<u>הסבר לפלט התוכנית</u>

הדוח מודפס בראש דף חדש, שבראשו יש שורת כותרת ומתחתיה קוי הדגשה:

<u>NØ.</u> <u>IDENT.</u> <u>NAME</u> <u>KØD</u>

NØ. - מספר סדורי של המצביע
IDENT. - מספר ת.ז. של המצביע
NAME - שם המצביע
KØD - מספר המועמד עבורו הצביע

בגוף הדוח שהוא תדפיס נתוני הקלט ישנן 21 שורות, ברווח כפול. ראה את קוד דילוג השורות בהוראת האיפיון 100 שבתוכנית.

לאחר הדוח המפרט את פרטי ההצבעה של כל משתתף מופיעה שורה המציינת את המועמד שזכה. זהו מועמד 2 שעבורו הצביעו 9 חברים. לשורה זו יש שורת הדגשה.

בסוף הדוח ישנן 5 שורות שבהם נרשמו יחסי הקולות של המועמד שזכה לשאר המועמדים.

כאן מסתיימת התוכנית ועמה גם הדוח.

ס י כ ו ם

בפרקים האחרונים תיארנו את התכונות הבסיסיות של השפה פורטרן. התיאור מספיק כדי שהקורא יוכל לכתוב תוכניות פשוטות ולהבין תוכניות הכתובות בפורטרן אולם אין הוא מכיל את כל האפשרויות וההוראות של השפה.

כפי שציינו במבוא כוונתנו היא לתת הכרה ראשונית וכללית של השפה, מעין סיור מהיר דרכה.

אנו מקווים כי פרקים אלו נתנו ללומד הבנה של המושג שפת תכנות בכלל, ושל האפשרויות הגלומות בשפת פורטרן בפרט.

חלק רביעי:

תכנות קובול

COBOL

פרק ו: שפת התכנות קובול

מאפייני השפה

השם COBOL בנוי מראשי התיבות של השם:

COmmon Business Oriented Language

ההגדרה של שפת קובול היא:

- שפת תכנות
- המשמשת לעיבוד נתונים מינהלי
- ואינה תלויה במבנה מסויים של מחשב.

נפרט את שלשת המאפיינים האלו:

א. שפת תכנות

שפת קובול שייכת לקבוצת שפות התכנות העיליות. השפות העיליות מיועדות להקל על עבודת התכנות על ידי שימוש במשפטים בניב של השפה האנגלית הברורים ככל האפשר למשתמש. המהדיר מתרגם את משפטי השפה לתוכנית בת-ביצוע (object program). לכל מחשב יש מהדיר מתאים אשר מפיק מתוכנית המקור, הנכתבת על ידי התוכניתן, את התוכנית אשר משתמשת בפקודות המיוחדות לו ועל כן יכולה לפעול בו בלבד.

קובול הינה שפה שמשפטיה דומים מאד לאנגלית וגם אדם ללא ידיעה מוקדמת יכול להבין משמעותם של משפטים אלה. כדוגמא נציג את משפטי קובול הבאים:

```
HEMSHECH. READ KØVETZ-AV AT END GØ TØ FINISH.
          IF GIL IS GREATER THAN 18 GØ TØ GIUS
          ELSE ADD 1 TØ MØNE-GADNA.
          ------------
GIUS.     WRITE RESUMAT-SHEM AFTER ADVANCING 2 LINES.
          ADD 2 TØ MØNE-SHURØT.
          IF MONE-SHURØT < 50 GØ TØ HEMSHECH.
          PERFØRM SUBRØUTINE-KØTERET.
          ------------
```

התוכנית מורכבת מאוסף של משפטים המסתיימים בנקודה (.).
קטע תוכנית זה כולל כמה פעולות: קריאת רשומה מקובץ, בדיקת שדה GIL, הדפסת שורה והחלטה על מעבר לדף חדש.

בזמן קריאת הרשומות מהקובץ נעשית בדיקה לסוף הקובץ ואז מסתעפים לקטע התוכנית FINISH. עורכים בדיקה של שדה GIL. אם הגיל גבוה מ-18 עוברים לקטע התוכנית GIUS, ואם לא - מוסיפים 1 למונה חברי הגדנ"ע, וממשיכים. בקטע התוכ-נית GIUS מדפיסים את הרשומה לאחר קידום הנייר ב-2 שורות. מוסיפים 2 למונה השורות, ובודקים אם הגיע ל-50. אם המונה קטן מ-50 - ממשיכים בקריאת רשומות ואם לא - מבצעים שגרת-כותרת הכוללת מעבר לדף חדש וכו'.

שים לב ששמות שדות, מונים וקטעי תוכנית נכתבו במִלים עבריות בתעתיק אנגלי (חופשי). אין כל חובה לנהוג כך, אך צורה זו נבחרה כדי להקל על הקורא המשתמש בשפה העברית.

ב. <u>שפה המיועדת לעיבוד נתונים מינהלי</u>

השפה כוללת הגדרות נתונים והוראות לביצוע תהליכי העיבוד השונים. מאלו יש להדגיש את המשפטים וההוראות לטיפול בשדות במבנים ובקשרים שונים, וכן את הוראות העריכה להדפסה. עיבוד נתונים מינהלי מאופיין ע"י פעולות קלט ופלט רבות ומגוונות הכוללות עריכה להדפסה לעומת פעולות חישוב ועיבוד פשוטות יחסית.

ג. <u>אי תלות במחשב מסוים</u>

תוכנית המקור הנכתבת בשפת קובול עוברת תרגום והופכת לתוכנית בת-ביצוע ע"י התוכנית המתרגמת (Compiler) המותאמת למחשב ולמערכת ההפעלה שבה היא תופעל.

לשפת קובול "ניבים" שונים אשר נקבעו ע"י היצרנים השונים של המחשבים. כל "ניב" מכיל את השפה הסטנדרטית הבסיסית אשר מועשרת באפשרויות נוספות המיו-עדות לנצל את התכונות המיוחדות של כל מחשב. כך, שהשפות קובול אשר מופעלות במחשבים של יצרנים שונים, ולפעמים אף אצל אותו יצרן מְחשבים, אינן זהות לחלו-טין. מכאן שאין אפשרות לקחת תוכנית-מקור אשר יועדה מלכתחילה להפעלה במחשב אחד ולהפעילה במחשב אחר ללא תקונים ושנויים מסוימים.

ככל שמצטמצמים בתכונות הסטנדרטיות הבסיסיות של השפה, השנויים הנדרשים מעטים יותר ואפשר לראות את התוכניות כבלתי-תלויות במחשב.

שפת הקובול המקובלת כיום בתעשית המחשבים הינה ANS COBOL, אשר מותאמת לסטנדרט הלאומי של ארה"ב (American National Standard), וגם היא, כאמור, אינה זהה לכל המחשבים ולמערכות ההפעלה.

מרכיבי השפה

מערכת התוים

להלן פירוט התוים המוכרים ע"י קובול:

ספרות	0-9	מקף או סימן חיסור	-
אותיות לטיניות (אותיות "גדולות" בלבד)	A-Z	כוכבית או סימן כפל	*
גרשיים	"	סימן חיבור	+
סימן שוויון	=	נקודה ופסיק	;
גרש	'	סוגר ימני	)
גדול מ-	>	סוגר שמאלי	(
קטן מ-	<	סימן דולר	$
פסיק	,	נקודה או נקודה עשרונית	.
קו נטוי או סימן חילוק	/	תו רווח (blank)	b̸

הסימן המקובל לסימון תו רווח בקידוד הוא b̸. בטופס המשובץ אין צורך לציין זאת אלא די להשאיר משבצת ריקה.

שדות המוגדרים כאלפנומריים יכולים להכיל תוים שונים מאלו שברשימה. מכיון שתוכן שדות אלפנומריים "שקוף" לשפת הקובול, תהיה זו אחריותו של המשתמש לכך שתהיה שרשרת הדפסה מתאימה במדפסת לשם הדפסתם.

אותיות עבריות

שפת קובול אינה מכירה באותיות עבריות והתוכניות נכתבות באלפבית הלטיני. עריכת דוח"ות בעברית והדפסתם נעשית ע"י כך שמציגים אות לטינית לכל אות עברית.

שיטה זו מקובלת לצורך הדפסת כותרות, קבועים וגם תוכן שדות בקלט, פלט ובקבצים. בעת ההדפסה יש לדאוג לכך שבמדפסת יהיו שרשרת או תוף המכילים את התוים העבריים במקום, או בנוסף ללטיניים.

להלן ייצוג האותיות העבריות ע"י אותיות לטיניות. שים לב שיש רק אותיות במבנה דפוס בגודל אחד (אין אותיות גדולות וקטנות):

א	&	מ	N
ב	A	ן	Ø
ג	B	נ	P
ד	C	ס	Q
ה	D	ע	R
ו	E	ף	S
ז	F	פ	T
ח	G	ץ	U
ט	H	צ	V
י	I	ק	W
ך	J	ר	X
כ	K	ש	Y
ל	L	ת	Z
ם	M		

הדפסה בעברית:

בהקשר לשימוש בשפה העברית במחשבים ישנו נושא נוסף הדורש טיפול: הפיכת כיוון השדה האלפבתי. בקלט מוזנים השדות האלפבתיים משמאל לימין, כלומר ...ג ב א, לפי כיוון התנועה של הכרטיס במנקבת, או לפי כיוון ההזנה באמצעים אחרים. לגבי האנגלית אין כל הבדל מכיון שזהו כיוון הקריאה הרגיל. בטכסט עברי יש צורך להפוך את כיוון האותיות בשדה לצורך הדפסה כדי שאפשר יהיה לקרוא את תוכנו. כלומר, יש להציגו כ-א ב ג... וכו'.

ישנן דרכים אחדות לטפל בנושא זה, למשל ע"י העברת השדה, תו תו, אל השדה המקבל. ברוב המקרים משתמשים בשגרות תכנות סטנדרטיות של המערכת. המתכנת מכין את השדה המקורי, ומוסר לשגרה את כתובתו ואת כתובת השדה שבו יתקבל הטכסט "ההפוך". בשעת החזרה מהשגרה מכיל השדה המקבל את הטכסט בצורה הרצויה, מימין לשמאל. נושא זה לא טופל בקורס. שים לב שקבועים בעברית אפשר לרשום מלכתחילה בכיוון הרצוי ואז אין צורך בהפיכה.

מלים שמורות

שפת הקובול משתמשת באוסף של מלים הנקראות "מלים שמורות" (reserved words) לצורך הגדרות והוראות. אסור להשתמש במלים אלו שלא למטרתן, כמו למשל כשמות של משתנים. עם זאת, שנוי קל במלה השמורה מאפשר שימוש רגיל שלה, כמו למשל CALL היא מלה שמורה אולם CALL1 מותרת בשימוש.

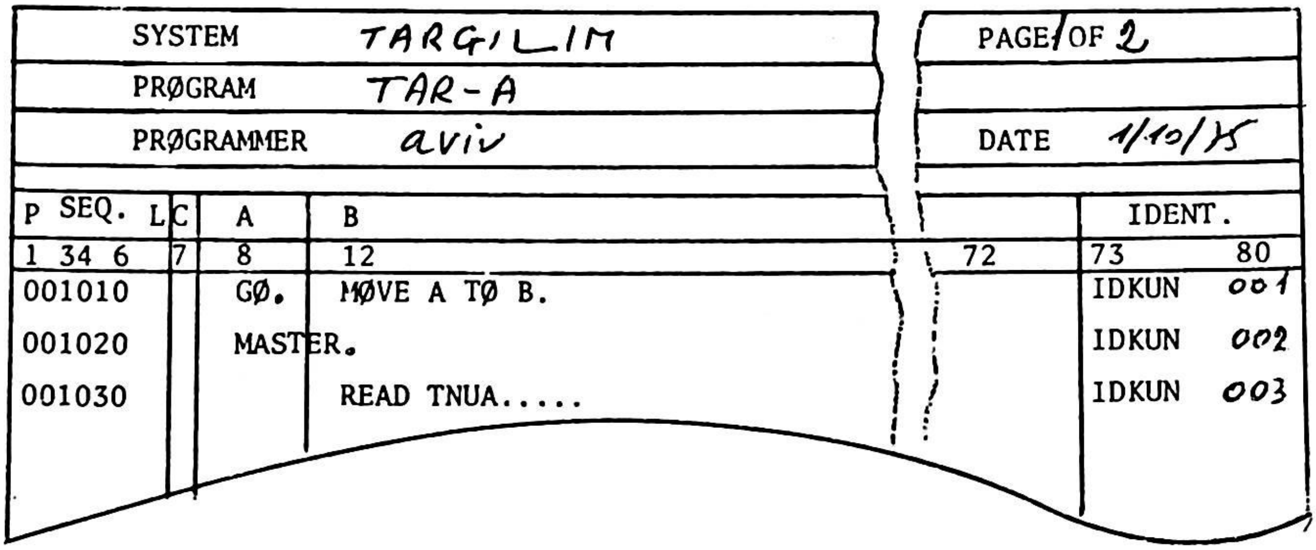
להלן לדוגמה, כמה מלים שמורות. רשימה מפורטת תמצא בספרות המתאימה ל"ניב" של שפת הקובול שבו תשתמש.

ACCESS	COMMA	ERROR
ADD	COMP	IF
AFTER	CONSOLE	MULTIPLY
AREA	COPY	SYMBOLIC
BEFORE	DATA	WRITE
BLOCK	DISPLAY	ZERO
CANCEL	END	
	EQUAL	

ככלל אפשר לראות כל מלה במבנה ההוראה כמלה שמורה של השפה. כלומר אי אפשר להגדיר שמות של שדות עם מלים שמורות.

טופס הקידוד

המתכנת בשפת קובול כותב את התוכנית בטופס במבנה מיוחד. מתוך הטופס מנקבים את התוכנית, שורה לכרטיס, ואת חבילת הכרטיסים שהיא התוכנית בשפת מקור - מזינים לתוכנית המתרגמת.



תאור הטופס:

א) 3 השורות הראשונות הן למידע בלבד ואינן מיועדות לניקוב.

ב) טורים 1-6 משמשים לספרור שוטף של שורות (כרטיסי) התוכנית. מילוי טורים אלו איננו נדרש ע"י השפה אך הוא חיוני לנוחיות עבודתו של המתכנת. ספרור השורות המומלץ הוא כדלקמן:

טורים 1-3 מספר דף הקידוד

טורים 4-5 מספר השורה בדף

בשיטה זו אפשר להשתמש בטור 6 להוספת כרטיסים בעתיד - עד 10 כרטיסים נוספים בין כל 2 כרטיסים מקוריים. כאשר רושמים טורים אלו, יבדוק המהדיר של שפת קובול את סדר הכרטיסים לפי מספרם הסידורי, ובהודעת השגיאה "יתריע" על סדר משובש.

ג) טור 7 משמש לסימון המשכיות של הגדרת תוכן שדה. כלומר, הוא מיועד למקרים בהם גולשת הגדרה כזו מעבר לתכולה של כרטיס אחד. צריך לסמן בו תו כלשהו.

ד) טורים 8-72 משמשים לכתיבת התוכנית. בטורים אלה קיימים שני אזורים:

אזור א (A) מתחיל בטור 8 (חובה להתחיל לרשום בטור 8).

אזור ב (B) מתחיל מטור 12 ואילך (אפשר להתחיל לרשום בכל טור באזור זה, החל מטור 12 ועד סופו - בטור 72).

החלוקה לאזורים נדרשת ע"י חוקי הכתיבה של קובול והשימוש בהם יוסבר בכל מקרה.

ה) טורים 73-80 משמשים לרישום זיהוי התוכנית ויהיו שוים בכל כרטיסי התוכנית. בטורים אלו ירשום המתכנת את שם התוכנית או את שמו למשל. לודגמה:
לסדרת התוכניות לניהול מלאי אפשר לרשום:

INVENT-A

INVENT-B

או, לתוכניות להנה"ח אפשר לרשום:

HN-KELET

HN-MIYUN

HN-IDKUN

חוקי הפרדה

לשפת תכנות, כמו לשפה רגילה, יש כללי כתיבה המאופיינים ע"י מבנה המשפטים, הסדר של המלים במשפט, סימני הפיסוק וכד'. חלק מלימוד השפה הוא הכרת חוקי ההפרדה:

אשר אותם יש לשנן היטב ולהקפיד בישומם. כל סטיה מכללי הכתיבה תגרום לשגיאה בתוכנית:

א. מרווח מפסיק בכל מקרה את המלה הכתובה לפניו, כלומר אינו יכול להיות חלק ממלה. לדוגמה: KOVETZ AV איננה מלה אחת בגלל המרווח המפריד בין המלה KOVETZ והמלה AV. אפשר לכתוב KOVETZAV או KOVETZ-AV, אם דרוש.

ב. לפני ואחרי כל סימן אריתמטי וסימן שויון (= + - * /) יבוא מרווח. לדוגמה: COMPUTE A = 1 + B - C. כשהסימן אינו מסמל פעולה חשבונית אלא סימנו של מספר, הוא ייכתב משמאל למספר וצמוד אליו, לדוגמה: +420 או -20.

ג. לפני פסיק (,) ונקודה (.) המופיעים במשפטי הקובול לא יהיה מרווח. אחריהם חייב להיות מרווח. לדוגמה: ... MOVE A TO B, MOVE C TO D. GO TO

כל משפט בשפת קובול מסתיים בנקודה והיא חייבת להופיע. פסיק אפשר לרשום כדי להפריד חלקי משפט לשם בהירות והוא אינו משפיע על משמעות המשפט מבחינת שפת קובול.

נקודה עשרונית ופסיק המופיע בשדה עריכה של מספר הינם יוצאים מהכלל ואחריהם לא יהיה מרווח. במקרה זה הם שייכים לרצף התוים.

ד. בכל מקרה שיש להשאיר מרווח אחד, אפשר להשאיר מספר כלשהו של מרווחים.

מבנה תוכנית

תוכנית קובול מורכבת מארבע חטיבות עיקריות (DIVISIONS) המופיעות בה בסדר קבוע כדלקמן:

1. IDENTIFICATION DIVISION

חטיבה זו מכילה נתוני זיהוי של התוכנית ושל המתכנת.

2. ENVIRONMENT DIVISION

חטיבה זו מתארת את האמצעים הנדרשים ע"י התוכנית:

- תאור המחשב
- תאור יחידות קלט/פלט

חטיבה זו מאחדת את כל הגורמים התלויים פיזית במחשב מסויים, ולכן כאשר מתכוונים להעביר את התוכנית להפעלה במחשב אחר יש לבצע כאן מספר שנויים.

3. DATA DIVISION

חטיבה זו מכילה את המידע שיעובד ע"י התוכנית:

- רשומות הקלט/פלט
- שדות עזר
- קבועים (constants)

4. PROCEDURE DIVISION

חטיבה זו מכילה את הפעולות הדרושות לעיבוד המידע, כלומר "הוראות" קובול.

תוכנית הקורס

קורס זה בקובול מיועד להקנות לקורא ידע בסיסי בשפה כדי שיוכל לכתוב ולהבין תוכניות פשוטות ובינוניות. מסיבה זו התרכזנו בעיקר בשיטות ארגון קבצים ובמשפטי העיבוד העיקריים ונמנענו מלהציג נושאים שבהם יש הבדלים בין "ניבי" שפת קובול השונים. מחשבה זו הנחתה בהשמטת נושא ניהול הנתונים בדיסקים מגנטיים, מכיון ש-נושא זה מורכב ומחייב ידע רב נוסף בשיטות ארגון נתונים בדיסקים. עם זאת - כל שנאמר לגבי סרטים מגנטיים יכול להתאים לשיטת ארגון סיקוונציאלי בדיסקים.

מומלץ שהקורא יפנה לספרות הכללית כדי למצוא את ההשלמות וההרחבות הדרושות להכרת השפה בשלימותה (ראה ביבליוגרפיה). המלצתנו לפנות לספרות זו רק לאחר נס-יון מעשי מסויים ברמה של השפה הנלמדת בקורס זה

במהלך ההסברים להכרת השפה ואפשרויותיה ניתנו דוגמאות ותרגילים שונים. חלק מהמקרים מהווה התרגיל השלמה לחומר הנלמד וכולל הסברים והבהרות החורגים מהסבר הדוגמה עצמה. הקורא מתבקש לשים לבו לכך. כמה מהתרגילים כוללים רק קטע מתכנות הבעיה, בכדי להקל על הקורא ולא להכביד עליו בפרטים לא רלוונטיים לנושא הלימוד. בנוסף, הוצגו מספר תרגילים בשלימותם כדי לאפשר לימוד ההצגה והתכנות של בעיה שלימה.

שפת קובול מתאימה במיוחד לתוכניות מינהליות ולכן רב בה החלק המוקדש להג-דרת נתונים, קבצים ודוח"ות. מאחר ואין אפשרות לבצע תוכנית כלשהי ללא הגדרות אלה, חלקו הראשון של הקורס מוקדש רובו ככולו להגדרת נתונים.

שיטת הכתיבה

להלן מספר הסברים על שיטת הכתיבה של הוראות קובול - כפי שתמצא בפרק זה: בהצגת המשפטים של השפה מופיעות לעתים בסוגריים מלים הכתובות עברית. מלים אלו תפקידן להראות לקורא מהו המרכיב שיש לכתוב במקומן בגוף המשפט. למשל:

GØ TØ (שם קטע)

במקום (שם קטע) ייכתב שם של קטע תכנות שיענה על החוקים המתאימים לשמות קטעים לדוגמה: .GØ TØ LØØP הסוגריים אינם חלק מהמשפט אלא באים לתחום את התאור העברי.

חלקי המשפטים שיתוארו בתוך סוגריים גדולים כפולים הינם חלקים שחובה לבחור באחת מהאפשרויות המוצגות בתוכם. סוגריים אלו אינם שייכים לשפת קובול, לדוגמה:

$$\text{BLØCK CØNTAINS n} \begin{Bmatrix} \text{RECØRDS} \\ \text{CHARACTERS} \end{Bmatrix}.$$

פירושו שהמשפט הנ"ל ניתן לכתוב בשני אופנים (וחובה לבחור באחד מהם):

BLØCK CØNTAINS n RECØRDS.

או BLØCK CØNTAINS n CHARACTERS.

סוגריים מרובעים אינם שייכים לשפת קובול, אלא מציינים חלק משפט שברשות כלומר, שאפשר להשמיטו מתוך המשפט. אפשר לכלול בסוגריים גם ברירה בין אחת מכמה אפשרויות:

COMPUTE FIELDZ [RØUNDED] = A + B.

בדוגמה: את המלה ROUNDED אפשר לכלול או להשמיט לפי הענין.

פרק 2: IDENTIFICATION DIVISION

תאור ומבנה

חטיבה זו מכילה את נתוני הזהוי של התוכנית ושל המתכנת, ויכולה להכיל פרטי זהוי והסבר נוספים.

המשפט IDENTIFICATION DIVISION הוא המשפט הראשון בתוכנית, ויכתב כשאר שמות ה-DIVISIONS באזור א (מטור 8). לאחריו תרשם נקודה.

שאר המשפטים ירשמו אחריו, אף הם באזור א כשלאחריהם נקודה. המשפטים הכלו-לים ב-DIVISION הם:

```
IDENTIFICATIØN DIVISIØN.
PRØGRAM-ID. 'program-name'.
AUTHØR. sentence.
INSTALLATIØN. sentence.
DATE-WRITTEN. sentence.
DATE-CØMPILED. sentence.
SECURITY. sentence.
REMARKS. sentence.
```

שם התוכנית - .PROGRAM-ID

זהו משפט חובה וחייב להיות המשפט השני בכל תוכנית קובול. המשפט מציין את שם התוכנית ומאפשר זהוי חד-ערכי שלה. הזהוי מיועד לשם אחסון התוכנית בספרית התוכניות ולשם הטענתה לזכרון המחשב לביצוע.

כללי הכתיבה של "שם התוכנית" הם:

- עד 8 תוים, ורק אלפבתיים ומספריים
- התו הראשון חייב להיות אלפבתי
- לפני ואחרי השם יבוא גרש (צמוד לתו הראשון והאחרון).

<u>שאר המשפטים</u>

שאר המשפטים הינם רשות, ולא חייבים להופיע בתוכנית. הם מספקים מידע כללי כמו שם המתכנת, מיתקן המחשב בו פותחה התוכנית, תאריך כתיבתה והרצתה של התוכנית, ונתונים כלליים אחרים.

אם כותבים אחד ממשפטים אלו, הוא ייבדק ע"י המהדיר ולכן חייב לענות על דרישות השפה:

- שם המשפט ייכתב באזור א (מטור 8) - ללא שגיאות כתיב.
- בסיום כל משפט תרשם נקודה ואחריה מרווח אחד לפחות.
- המשפט בהמשך ייכתב בטכסט חופשי כרצון המתכנת, אך חייב להסתיים בנקודה.

<u>דוגמה:</u>

```
A   B
8   12
IDENTIFICATIØN DIVISIØN.
PRØGRAM-ID. 'TØCHNIT1'.
AUTHØR. ALMØNI.
DATE-WRITTEN. 01/04/74.
REMARKS. FØR MØRE INFØRMATIØN CALL FIVE-Ø.
```

פרק 3: ENVIRONMENT DIVISION

כללי

חטיבה זו בתוכנית קשורה קשר הדוק עם זהוי יחידות קלט פלט והגדרת קבצים. הוראות חטיבה זו יפורטו עם תאור נושא הקבצים מאוחר יותר.

בשלב הראשון של הצגת התוכנית בקובול נשתמש באמצעי קלט/פלט ובשיטות הזנה והפקה של נתונים שאין צורך להגדירם במפורש. נשתמש בדרך כלל בקלט כרטיסים וב-פלט מכונת הכתיבה של המפעיל. פלט בשיטה זו טוב להודעות בלבד ולא להפקת דווחים. למכונת הכתיבה למשל אין אמצעים לשליטה על קידום הנייר כפי שנדרש במדפסת ומהי-רותה מתאימה לקצב עבודת אדם. קבצים בנפח גדול נמצאים בסרטים ובדיסקים מגנטיים ואלו דורשים הגדרות מפורשות.

יתרונה של ההצגה הזו בכך שהתלמיד יכול להתרכז בלימוד מבנה הנתונים וההו-ראות של השפה מבלי לטפל בהגדרת הקבצים שהיא בעיה טכנית בעיקרה. כמו-כן מאפשרת שיטה זו לפשט את הצגת הפתרון לבעיות השונות הניתנות במהלך הקורס.

הגדרת יחידות קלט ופלט וקבצים תלויה גם במחשב ובמערכת ההפעלה שבה תופעל התוכנית, ולכן יש לפנות לקבלת הסברים נוספים אל הספרות המתאימה.

פרק 4: DATA DIVISION

תאור ומבנה

ה-DATA DIVISION מתאר את הנתונים שיעובדו ע"י התוכנית. נתונים אלו כו-
ללים:

א. נתונים הנכללים ברשומות קלט/פלט הנכנסות או יוצאות מהזכרון של המחשב
ב. נתוני ביניים המתקבלים בתהליך עיבוד הנתונים (לדוגמה: תוצאות ביניים של חישוב מסויים)
ג. קבועים (קונסטנטות) שהוגדרו ע"י המשתמש.

ה-DATA DIVISION מתחלק למספר פרקים - SECTIONS:

א. FILE SECTION הוא הפרק בו מתוארים קבצי הקלט/פלט המשתתפים בתוכנית:
תאור הרשומה, מבנה הגוש, תויות וכד'.
פרק זה יפורט מאוחר יותר, בנושא קלט/פלט.

ב. WORKING-STORAGE SECTION הוא הפרק בו מוגדרים שדות עזר, קבועים ואף רשומות.

המשפטים המתארים את ה-DIVISION וה-SECTION ייכתבו באזור א, עם נקודה בסופם.

```
|A   |B
|8   |12
|DATA DIVISIØN.
|FILE SECTIØN.
|WØRKING-STØRAGE SECTIØN.
```

תאור שדות - WORKING-STORAGE SECTION

תאור שדות כולל את המרכיבים הבאים:

א. רמת השדה (level number)
ב. שם השדה
ג. מבנה השדה.

רמת השדה

שדות יכולים להיות שדות עצמאיים העומדים בפני עצמם (כמו קבועים או מונים) ויכולים להיות שדות המאוגדים בשדה אחר או ברשומה. לדוגמה: רשומה של קובץ התו-שבים מכילה את השדה "כתובת" של כל אזרח, מכאן שהשדה כתובת הוא חלק מתוך שדה גדול יותר שהוא הרשומה. שדה "כתובת" יכול לאגד בתוכו מספר שדות אחרים כמו שם רחוב, מספר בית, עיר ומיקוד.

יחסים אלה בין השדות (שדה מאגד, שדות מאוגדים) מתוארים ע"י מספרים שנק-ראים רמות השדה (level numbers).

כלל היחסים בין השדות הוא: שדה בעל מספר רמה גדול יותר כלול (או מאוגד) בשדה אחר שמספר רמתו נמוך יותר (מספר קטן יותר) ושהוגדר לפניו. במלים אחרות אפשר לומר ששדה עם מספר רמה גבוה יותר כלול בשדה אחר, או מהווה חלק משדה אחר שמספר הרמה שלו נמוך יותר.

לדוגמה: כתובת - 03 ADDRESS
מס' בית - 04 NUMBER
שם רחוב - 04 STREET

השדה ADDRESS מאגד שני שדות STREET,NUMBER מכיון שמספר הרמה שלו נמוך תר משלהם והם רשומים אחריו. השדות STREET,NUMBER הם בעלי רמה שוה וכלולים יחד בשדה ADDRESS.

נדגים את מבנה הרשומה ורמות השדות ע"י תאור רשומת פרטים אישיים בקובץ תושבים:

רשומת תושב						
תאריך לידה	כתובת			שם תושב		מספר זהות
	רחוב	מספר בית	מיקוד	שם פרטי	שם משפחה	

א. ניתן להתיחס לכל הרשומה כאל שדה אחד שתוכנו נתוני התושב.

ב. את הרשומה ניתן לחלק (תת-חלוקה) לארבעה שדות:

מספר זהות, שם התושב, כתובת, תאריך-לידה.

כל אחד מארבעת השדות הינו חלק מהשדה ששמו "רשומת תושב" ומהווה תת-חלוקה שלו. לכן מספרי רמת השדות האלו יהיו גבוהים מרמת-השדה של רשומת התושב (כלומר, מספר גדול יותר).

השדה "שם התושב" אינו כלול בשדה "מספר זהות" ואינו מהווה תת-חלוקה שלו ולכן מספרי רמות שני השדות האלו יהיו זהים (אותו מספר).

ג. השדה "שם תושב" מתחלק בחלוקת משנה לשדות "שם פרטי" ו-"שם משפחה" המהווים חלק ממנו. מכאן שמספר רמת השדות "שם פרטי" ו"שם משפחה" יהיו גבוהים מאלה של "שם תושב".

ד. באופן אנלוגי יהיו מספרי רמת השדות "רחוב", "מספר בית" ו"מיקוד" גבוהים ממספר רמת השדה "כתובת".

אם נציין את מספרי רמת השדות במספרים נמצא שלשדות הנ"ל מתאימים המספרים הבאים:

רמה 1 - שדה "רשומת תושב"

רמה 2 - שדות "מספר זהות", "שם תושב", "כתובת", "תאריך לידה".

רמה 3 - שדות "שם משפחה", "שם פרטי", "מיקוד", "מספר בית" ו"רחוב".

אפשר לתאר את הרשומה בצורה הבאה (תאור מדוייק בקובול ראה בהמשך).

תושב	1
מספר זהות	2
שם תושב	2
שם משפחה	3
שם פרטי	3
כתובת	2
מיקוד	3
מספר בית	3
שם רחוב	3
תאריך לידה	2

שדה שמחולק לשדות אחרים ("מאגר") נקרא <u>נתון קבוצתי</u> - Group Item לדוגמה:

"כתובת". שדה שאינו מחולק לשדות אחרים נקרא נתון אלמנטרי - Elementary Item,
לדוגמה: "מיקוד".

השדות מוגדרים בקובול זה לאחר זה, כל אחד בשורה נפרדת. שדה שמספר רמתו גבוה יציין תמיד תת-חלוקה של הנתון הקבוצתי בעל מספר הרמה הנמוך ממנו שהוגדר לפניו (דוגמה ראה להלן - בסעיף הבא).

מספר הרמות החוקיים בקובול הם:

מספר 01 - מספר הרמה הנמוך ביותר המוקדש להגדרת הרשומה השלימה.
מספרים 02-49 - משמשים להגדרת שדות שמהווים תת-חלוקה של שדה אחר.
המספר 77 - מוקדש לנתונים אלמנטריים עצמאיים, כלומר שאינם מהווים חלק מנתון קבוצתי (לדוגמה: קבוע או שדה המשמש כמונה).

בחירת המספרים בין 02-49 הינה חופשית. אין חובה לבחור במספרים עוקבים, אך יש לשמור על כללי חלוקת השדות ע"י ציון מספר גבוה יותר כשנדרש בלבד.

את כל השדות המוגדרים ברמה 77 (נתונים אלמנטריים) יש להגדיר לפני הגדרת שאר הרמות, מיד בתחילת ה-WORKING-STORAGE SECTION של ה-DATA DIVISION.

שם השדה

שם השדה (או "שם משתנה") ניתן ע"י המתכנת כדי שיוכל להתיחס לאותו שדה בעת עיבודו. שם השדה מתיחס לשדה עצמו ולא לתוכנו ברגע מסויים, כך שאותו שם יכול להתיחס לערכים שונים ככל שאלה יתחלפו בתוך השדה במהלך העיבוד. לדוגמה שדה A יכול להכיל בתחילת העיבוד את הערך 5 ולאחר מכן את הערך 168.

שם שדה יכול להיות מורכב מאותיות, ספרות ומקף וחייב להכיל לפחות אות אחת.

לדוגמה: SHEM-PRATI, NAME - שם שדה חוקי
AB2345 - שם שדה חוקי
12457 - שם שדה לא חוקי כי אינו מכיל אות בראשו.
MØVE - שם שדה לא חוקי כי זו מלה שמורה של הקובול
SHEM MISHPAHA - שם שדה לא חוקי כי מכיל מרווח

שם השדה מוגדר לאחר מספר הרמה, כשמרווח אחד לפחות מפריד ביניהם. אין להשתמש באותו שם ליותר מאשר שדה אחד, מכיון ששם השדה צריך להיות חד-משמעי. את השם אפשר לכתוב כמלה לועזית או בתעתיק של המלה העברית, כפי שנהגנו ברוב המקרים.

אם יש צרוך להגדיר שדה אך אין התיחסות אליו בתוכנית (למשל חלק מרשומה שאין לה שימוש בתוכנית מסויימת) אפשר לתת לו את השם FILLER. שם זה הינו שם כללי פיקטיבי שמאפשר הגדרת גודל ומבנה השדה אך לא מאפשר פניה אליו. מותר להש-תמש בשם זה יותר מפעם אחת בתוכנית.

<u>לדוגמה</u>: הגדרת רשומת התושב המתוארת בסעיף הקודם:

```
A  B
01 RESHUMA.
   03 MISPAR-ZEHUT.
   03 SHEM-TØSHAV.

         07 SHEM-MISHPAHA.
         07 SHEM-PRATI.

   03 KTØVET.

         05 MIKUD.
         05 MISPAR-BAIT.
         05 REHØV.

   03 BIRTH-DATE.
```

<u>שים לב</u>:

א. הרמה 01 נכתבת באזור א. שאר הרמות באזור ב (החל מטור 12) בכל טור שהוא.

ב. רישום הרמות הגבוהות בטופס בצורה מדורגת (הזזה ימינה) אינה הכרחית, אך היא נוחה ומאפשרת לראות ביתר קלות את חלוקת השדות.

ג. החלוקה של השדהSHEM-TOSHAV נעשתה ברמה מס' 07 ואילו חלוקת שדה KTOVET נעשתה ברמה מס' 05. עובדה זו אינה מפריעה כלל כל עוד נשמר הכלל שנתונים אלמנטרים יהיו במספר רמה גבוה מהנתון הקבוצתי שלהם.

ד. השדה SHEM-TOSHAV אינו כלול ב- MISPAR-ZEHUT ולכן אסור שמספר רמתו יהיה גבוה יותר. אך מותר היה לקבוע ל- SHEM-TOSHAV רמה 02 כל עוד מספר זה גבוה מהנתון הקבוצתי שלו (במקרה זה RESHUMA) ולא גבוה מהנתון שלפניו שאינו נתון קבוצתי שלו.
אם היינו קובעים ל-SHEM-TOSHAV מספר רמה 02 חייבים היינו לשנות גם רמתם של השדות KTOVET ו-BIRTH-DATE ל-02 כי אם לא נעשה כן מספר הרמה שלהם יהיה גבוה יותר מרמת SHEM-TOSHAV, כלומר שהם יוצגו כתת-חלוקה שלו, מה שאינו תואם את המציאות.

ה. בסוף כל משפט הגדרת שדה יש נקודה הצמודה אל האות האחרונה של המשפט.

מבנה השדה

בתאור מבנה השדה קובעים את הנתונים הבאים:

- אופי השדה
- אורך השדה

א. אופי שדה

אופיו של שדה יכול להיות אחד מהבאים:

מספרי - שדה שיכיל מספרים בלבד ("נומרי").
אלפביתי - שדה שיכיל אותיות, מקפים ומרווחים בלבד.
אלפנומרי - שדה שיכיל כל תו שהוא

יש לקבוע במדוייק אופיו של כל שדה בגלל הסיבות הבאות:

1) אם השדה הוגדר כאלפביתי או מספרי ובזמן ההתיחסות לאותו שדה בעת הביצוע של התוכנית, תוכנו הרגעי אינו מתאים לאופי שהוגדר, הדבר יגרום להפסקת הרצת התוכנית תוך ציון שגיאה מתאימה. למשל, מכיון שמבצעים פעולה חשבונית עם אותיות וסימנים.

2) שדה שמשתתף בפעולה חשבונית כלשהי חייב להיות מוגדר כמספרי - אחרת מתקבלת הודעת שגיאה בעת הידור התוכנית (compilation).

3) חוקי העברת נתונים משדה לשדה, באיפן כללי, הם:
- לשדה אלפנומרי אפשר להעביר שדה מכל סוג.
- לשדה מספרי אפשר להעביר רק שדה מספרי
- לשדה אלפביתי אפשר להעביר רק שדה אלפביתי

אופיו של שדה מוגדר על ידי תוים מוסכמים כדלהלן:

האות A מציינת אופי אלפביתי

האות X מציינת אופי אלפנומרי

הספרה 9 מציינת אופי מספרי.

. ב. <u>אורך השדה</u>

אורכו של שדה נקבע על פי מספר התוים <u>המירבי</u> שניתן לאחסן בו, והוא יוגדר ע"י מספר התוים המוסכמים המתארים את אופי השדה.

לדוגמה: שדה אלפנומרי שאורכו 5 תוים יוגדר ע"י XXXXX.

שדה מספרי שאורכו 3 ספרות יוגדר ע"י 999.

<u>הערה:</u>

הגדרת השדות המספריים המוצגת כאן מתיחסת לשדות מספריים מסוג DISPLAY. אלו הם שדות שמכילים מספרים במבנה הקלט או מספרים מוכנים להדפסה, כלומר כל ספרה מוצגת בזכרון כיחידה נפרדת לפי צרוף הסביות המתאים שלה מבלי לעבור הסבה לשיטת המספרים המקובלת במחשב.

ברוב המחשבים לא נעשית פעולת החישוב במספרים עשרוניים אלא באחת משיטות הייצוג: השיטה הבינרית, השיטה של הנקודה הצפה (floating point) ושיטת הדצימלי המקובץ (packed decimal). כאשר מבצעים פעולת חישוב בשדה עשרוני "חיצוני" - מבוצעת באופן אוטומטי, ע"י הקובול, הסבת השדה לשיטת הייצוג הנהוגה במחשב והחזרת התוצאה למבנה עשרוני כפי שנדרש בהגדרה. כדי למנוע את בזבוז הזמן הכרוך בהפיכה זו הנעשית פעמים רבות בתוכנית, ומסיבות נוספות, מוחזקים המספרים בקבצים בשיטה השונה מהשיטה העשרונית.

בנושא זה - עיין בספרות המתאימה לניב הקובול המתאים למחשב שאתה מקבל את שרותיו.

קיימים שדות מספריים מסוגים נוספים, אשר מיועדים לביצוע פעולות חשבוניות:

1) <u>שדות בינריים</u> - שדות אלה אינם כוללים ספרות עשרוניות אלא מספרים בינריים, כלומר מספרים המורכבים מהספרות 0 ו-1. ערך השדה מתקבל ע"י חזקות של הבסיס 2.

2) <u>שדות מסוג הנקודה הצפה</u> - שדות אלה מוצגים ע"י שבר הקטן מ-1 (מנטיסה) ומע-

ריך חזקה (אקספוננט). ערך המספר מתקבל ע"י מכפלת הבסיס המועלה לחזקה במנטיסה.

לדוגמה: המספר 63 יכול להיות מוצג ע"י 0.63×10^2. כש-0.63 הוא המנטיסה ו-2 - האקספוננט.

מספרים אלה מאפשרים תצוגת מספרים גדולים מאד או מספרים קטנים מאד (ע"י שי-מוש באקספוננט שלילי).

3) <u>שדות מסוג דצימלי מקובץ</u> - שדות אלה מנצלים את העובדה שכל ספרה עשרונית ניתנת להצגה בארבע סביות (bits). בשדות אלה מוחזקות 2 ספרות בכל בית בזכרון (8 bits ≡ byte) כשהסימן מופיע בסוף המספר כלומר בחצי הימני של הבית הימני ביותר (4 סביות).

הגדרת מבנה השדה - PICTURE

<u>הגדרה בסיסית</u>

הגדרת מבנה השדה נעשית על פי כללי השפה בעזרת מלת המפתח <u>PICTURE</u> או <u>PIC</u>. מלת המפתח תופיע <u>לאחר</u> שם השדה <u>ולפני</u> הגדרת אופיו של השדה וגודלו.

לדוגמה, שדה אלפנומרי בגודל 8 תוים ובשם SADE יוגדר כך:

$$77 \text{ SADE } \left\{ \begin{array}{l} \text{PICTURE} \\ \text{PIC} \end{array} \right\} \text{ XXXXXXXX.}$$

כלומר, אפשר לכתוב PICTURE <u>או</u> בקיצור PIC (<u>ללא</u> סוגריים).

כאשר השדה ארוך ומכיל יותר מתו אחד, אפשר להשתמש בשיטת "מקדם הכפלה". מקדם-ההכפלה הינו מספר המופיע בסוגריים <u>צמוד</u> לתו המציין את אופי השדה ומורה את מספר הפעמים שהתו צריך להופיע.

לדוגמה: XXXX אפשר לרשום כ-X(4)

999 אפשר לרשום כ-9(3)

או: 77 SADE PIC X(8).

<u>תרגיל:</u>

יש להגדיר בתוכנית 2 קבועים מספריים באורך של 10 ספרות כל אחד בשם MONE 1, MONE 2 בהתאמה. יש להגדיר גם רשומה לפתיחת חשבון בנק במבנה הכא:

מספר לקוח - שדה מספרי באורך 8 ספרות.
שם הלקוח - מחולק ל-
שם פרטי - אלפביתי באורך 10 תוים
שם משפחה - אלפביתי באורך 10 תוים
פרטי פתיחת חשבון - שני שדות משנה: שם הפקיד הרושם ותאריך
שם הפקיד הרושם - אלפנומרי באורך 15 תוים
תאריך מחולק ל-
יום - 2 ספרות
חודש- 2 ספרות
שנה - 2 ספרות

תאור גרפי של הרשומה:

<table dir="rtl">
<tr><td colspan="7">רשומת לקוח</td></tr>
<tr><td colspan="4">פתיחת חשבון</td><td colspan="2">שם לקוח</td><td rowspan="3">מספר
לקוח</td></tr>
<tr><td colspan="3">תאריך</td><td rowspan="2">שם
פקיד</td><td rowspan="2">משפחה</td><td rowspan="2">פרטי</td></tr>
<tr><td>שנה</td><td>חודש</td><td>יום</td></tr>
</table>

קטע התוכנית המתארת את הרשומה:

```
A   B
8   12
IDENTIFICATIØN DIVISIØN.
PRØGRAM-ID. 'TARGIL1'.

ENVIRØNMENT DIVISIØN.
    -
    -
    -
DATA DIVISIØN.
WØRKING-STØRAGE SECTIØN.

77  MØNE1   PIC 9(10).
77  MØNE2   PIC 9(10).

01  RESHUMA.

    03   MISPAR-LAKUAH   PIC 9(8).

    03  SHEM.
         06  PRATI        PIC A(10).
         06  MISHPAHA     PIC A(10).

    03  PTIHA.

         04  PAKID        PIC X(15).
         04  TAARIH.
                09 YØM    PIC 99.
                09 HØDESH  PIC 99.
                09 SHANA PIC 99.
```

שים לב:

א. שדות עם נתון אלמנטרי שאינם חלק מנתון קבוצתי הוגדרו ברמה 77 לפני שאר הרמות.

ב. בנתונים קבוצתיים ירשם מספר רמת השדה ושם השדה בלבד. אסור לרשום את תאור אופי השדה ומבנהו.
נתון קבוצתי (כלומר, שדה המתחלק לשדות אחרים) יהיה לעולם בעל אופי של שדה אלפנומרי ואורכו ייקבע על פי סה"כ אורכי השדות שכלולים בו בתת-החלוקה.
למשל: השדה TAARIH יהיה אלפנומרי (למרות שכל השדות הכלולים בו מספריים) ואורכו יהיה 6.
השדה PTIHA יהיה אלפנומרי ואורכו יהיה 21.
השדה RESHUMA יהיה אלפנומרי ואורכו 49.

ג. בסוף כל משפט הגדרה יש נקודה.

הגדרה של שדות מספריים

א. שמירת הסימן של שדה מספרי

שדה מספרי שהוגדר ע"י מספר תשיעיות (כמספר הספרות של השדה), כפי שתואר לעיל, יכיל תמיד מספר בערכו המוחלט. מכאן נובע שאם נעביר אל שדה זה מספר שלילי, הסימן ייעלם וערכו של השדה יהיה למעשה חיובי.

כדי לשמור על סימנו של המספר הנמצא בשדה יש להוסיף את האות S לפני התשיעיות וסמוך לספרה השמאלית. האות S הנוספת אינה מגדילה את השדה.
דוגמה: מספר בן ארבע ספרות עם שמירת סימן, המספר יוגדר כדלהלן:

```
77 SADE    PICTURE S9999.
```

ב. קביעת מיקומה של נקודה עשרונית

אם המספר המוגדר אינו מספר שלם, יש לציין מיקומה של הנקודה העשרונית. ציון מיקום זה מסייע בביצוע נכון של פעולות חשבוניות בהם נוטל חלק המספר המוג- דר. הנקודה העשרונית עצמה איננה נמצאת בשדה המספרי השמור במחשב ולכן מציינים למעשה את מיקומה הלוגי בשדה - היא נקודה עשרונית מדומה.

ציון המיקום ייעשה ע"י האות V. האות V, כדוגמת האות S אינה מגדילה את השדה ואינה באה "על חשבון" ספרה של המספר ("התשיעיות").

<u>לדוגמה:</u> הגדרת מספר בן 6 ספרות, מהן שתיים מימין לנקודה העשרונית, עם שמירת הסימן של המספר:

```
77 MISPAR  PIC  S9999V99.
77 MISPAR  PIC  S9(4)V99.        :או
77 MISPAR  PIC  S9(4)V9(2).      :או
```

ז כ ו ר!

שפת קובול אינה מכירה בנקודה עשרונית המופיעה כתו בשדה מספרי. אחסנת הנתונים בזיכרון ובקבצים הינה ללא הנקודה ורק המקום הלוגי שלה נשמר בתאור השדה. אם בקלט ישנם נתונים מספריים עם נקודה עשרונית (כתו), יש לבטל אותה <u>לפני</u> ביצוע כל פעולת חישוב ועיבוד בשדה זה. כאשר מתכוונים להדפיס שדה מספרי אשר <u>חייב</u> להכיל נקודה עשרונית - יש לערוך אותו במיוחד למטרה זו. לאחר העריכה אין זה יותר "שדה מספרי" ואסור לערוך בו פעולות חישוב

<u>עריכת שדות מספריים להדפסה</u>

מבנה שדות אלפנומריים בשורת ההדפסה זהה למבנה השדה בזיכרון המחשב וב-קבצים. בשדות מספריים אין הדבר כך, וזאת מפאת השוני שקיים בין מבנה הנתונים המספריים לביצוע פעולות חישוב ועיבוד,לבין מבנה אותם הנתונים בהדפסה.

נתונים מספריים במחשב יכולים להרשם בשיטות מספריות שונות: השיטה הדצי-מלית במבנה תוים, שיטה דצימלית מקובצת, שיטה בינרית ושיטת הנקודה הצפה. בהד-פסת שדות מספריים צריך קודם כל להציג את השדה בצורה דצימלית כפי שנדרש ע"י המדפסת. השלב השני הוא "עריכת השדה" (editing) לשם:

א. החלפת אפסים מובילים במרווחים (לדוגמה הדפסת 5 ולא 005).
ב. הכנסת נקודה עשרונית במקום הנכון, בהתאם להגדרת המספר.
ג. ציון סימן שלילי אם תוכן השדה הוא מספר שלילי

עריכת שדה מספרי נעשית ע"י העברת השדה המיועד לעריכה אל שדה אחר שהוגדר במיוחד לעריכה (Report item). העברת השדה תעשה ע"י הוראת העברה (MOVE) רגילה

כשהשדה המקבל יהיה שדה העריכה אשר מבנהו מוגדר בצורה מיוחדת לצורך זה. לאחר ההעברה, השדה השולח אינו משתנה (כרגיל) והשדה המקבל יכיל את המספר הערוך להד- פסה.

השדה הערוך איננו שדה מספרי (כי הוא כבר מכיל מרווחים וסימנים מיוחדים כמו נקודות, פסיקים וסימן המינוס) ולכן אסור לעשות בו חישובים או השוואות ארית- מטיות.

הגדרת שדה עריכה

שדה עריכה מגדירים בעזרת ההוראה PICTURE כפי שמגדירים שדות אחרים. בשדות עריכה קיימים תוים נוספים, מיוחדים, שמטרתם לציין את דרישות העריכה בשדה. את התוים מציינים בצורה הבאה:

א. תו Z

יבוא במקום תו 9 ויציין הפיכת אפסים מובילים למרווחים. מאחר והאפסים המובילים נמצאים משמאל למספר, יופיע גם התו Z תמיד משמאל לתו 9 הראשון. אפסים שיופיעו מימין לספרה שאינה אפס לא ייהפכו למרווחים כי אינם אפסים מובילים.

דוגמה: 77 SADEA PIC 9(4).

77 SADEB PIC ZZZ9.

SADEA מכיל את המספר 0023.

אחרי ההוראה של העברת SADEA ל-SADEB יהיה תכנו של SADEB: bb23.

להלן מספר דוגמאות נוספות:

תוכן SADEA	תוכן SADEB לאחר ההעברה
1000	1000
0130	b130
0100	b100
0000	bbb0

שים לב: מכיון שצויין 9 אחד במבנה SADEB, מימין, נשאר אפס אחד במקום זה ולא הפך לתו-רווח (b).

ב. נקודה עשרונית (.)

תצויין במקום הרצוי, לפי ערכו של המספר, והיא תופשת תו בשדה העריכה בזיכרון ובשורת ההדפסה. בזמן ההעברה של שדה מספרי אל שדה העריכה נעשית התאמה בין הנקודה העשרונית למעשה לבין הנקודה הלוגית המצויינת על ידי האות V בשדה השולח.

לדוגמה: שדה SADEA הוגדר כ-9(4)V99, כלומר 4 ספרות לשלימים ו-2 ספרות ימניות עשרוניות.

בטבלה שלהלן ניתנות מספר דוגמאות להעברת SADEA אל שדה עריכה SADEB.

	תוכן SADEA	הגדרת שדה עריכה SADEB	תוכן SADEB לאחר ההעברה
(1	400036	9(4)	4000
(2	003040	Z(4)	bb30
(3	000020	ZZZ9.9	bbb0.2
(4	001044	ZZZ9.9	bb10.4
(5	010240	ZZZ9.99	b102.40
(6	034567	ZZZ9.999	b345.670
(7	000004	ZZZZ.Z9	bbbbbb4
(8	000000	ZZZ.ZZ	bbbbbbb

שים לב: - Z לא יצויין לעולם מימין ל-9.
- אם התו Z צויין מימין לנקודה העשרונית וישנם אפסים מובילים עד הנקודה העשרונית, גם הנקודה תתורגם למרווח (דוגמאות 8,7).
- הגדרת SADEB קובעת את מספר הספרות שימין לנקודה העשרונית. כאשר יש שוני בין מספר הספרות העשרוניות בשדה זה וב-SADEA, העריכה מו-סיפה אפסים או משמיטה ספרות בהתאם למקרה (שורות 1-6,4).

ג. סימן המינוס (-)

יהיה התו הימני ביותר של הגדרת השדה במשפט ה-PICTURE. חוקי פעולתו הם הבאים:

- אם השדה שלילי, הסימן יופיע בהדפסה.
- אם השדה חיובי, הסימן יוחלף במרווח.

דוגמה: SADEA מוגדר 9(4)
SADEB מוגדר ZZ99- (5 תוים בשדה)

	תוכן SADEA	תוכן SADEB לאחר ההעברה
(חיובי)	0100	b100b
(שלילי)	0100	b100-

ד. פסיק (,)

משמש להפרדת ספרות האלפים במספר. חוקי פעולתו של הפסיק בשדה העריכה הם:
- אם משמאל לפסיק הוחלפו אפסים במרווחים, יהפך גם הפסיק למרווח.
- אם משמאל לפסיק יש ספרה השונה מאפס, ישאר הפסיק במקומו.

דוגמה: SADEA מוגדר 9(4)
SADEB מוגדר Z,ZZ9 (בשדה 5 תוים)

תוכן SADEA	תוכן SADEB לאחר ההעברה
0030	bbb30
1400	1,400
0460	bb460

בדוגמאות הראשונה והשלישית הפך הפסיק לתו b.

קיימים תוים נוספים לעריכת שדות מספריים, ועליהם יש ללמוד בספרות המתאימה.

הגדרת שדה עם ערך התחלתי - VALUE

ההגדרות שבהן טיפלנו עד עתה היקצו לכל שדה שטח מתאים בזיכרון. תוכן השדות לפני הפעלת התוכנית איננו ידוע ואין להניח לגביו דבר. כך למשל, שדה שישמש למונה חייבים לאפס לפני פעולת הסיכום הראשונה. לעתים יש צורך בהגדרת שדות עם ערך התחלתי כמו קבועים שונים, מונים מאופסים, כותרות לדוח"ות ועוד.

הגדרת תוכן התחלתי לשדה נעשית ע"י הוספת המלה VALUE לאחר תאור המבנה

(PICTURE) של אותו שדה:

77 MØNE PIC 99 VALUE (תוכן השדה)

למשל: 77 MØNE PIC 99 VALUE 27.

משפט זה מציין שערכו ההתחלתי של השדה MONE בן 2 ספרות הוא 27.

הגדרה בסיסית

את תוכן השדה מציינים בצורה שונה לשדות מסוגים שונים:

א. שדות מספריים

הכללים בקביעת תוכן התחלתי לשדה מספרי הם:

- תוכן השדה יכול להכיל ספרות (0-9), נקודה עשרונית וסימן אריתמטי בלבד, כל תו מאלו נרשם בטור נפרד בטופס הקידוד ויונקב בטור נפרד בכרטיס.
- הסימן האריתמטי יצויין משמאל למספר וצמוד לו. סימן + אין צורך לכתוב. סימן - יכתב תמיד
- אם מספר הספרות קטן מגודל-השדה יוספו אפסים משמאל לספרות השלימים ומימין לנקודה העשרונית, כדי לשמור על ערכו הנכון של השדה בהתאם להגדרת מבנהו.
- אם מספר הספרות גדול מגודל-השדה, יקוצצו ספרות שמאליות ביותר שמשמאל לנקודה העשרונית וספרות ימניות ביותר שמימין לנקודה העשרונית, בהתאמה.
- אם צויינה נקודה עשרונית, תעשה התאמת ספרות בין נקודה זו לאות V שצויינה במבנה ה-PICTURE אם הנקודה העשרונית אינה מסומנת, מניחים שהיא רשומה מימין לשדה המוגדר.

דוגמאות

הגדרה	הערך המתקבל בהתאם להגדרת PIC
77 SADE PIC 9(4) VALUE 1000	- 1000
77 SADE PIC 9(4) VALUE - 1000	+ 1000
77 SADE PIC S9(4) VALUE - 1000	- 1000
77 SADE PIC S9(4) VALUE 43	+ 0043
77 SADE PIC S9(4)V99 VALUE 10	+ 0010 00
77 SADE PIC S9(4)V99 VALUE 100 04	+ 0100 04
77 SADE PIC S9(4)V99 VALUE 2 4.	+ 0002.40
77 SADE PIC S9(4)V99 VALUE 123456	+ 3456 00

שים לב שהערך המתקבל אינו שוה תמיד לערך הרשום בהגדרת VALUE, מכיון שערך זה אינו תואם את הגדרה השדה ב- PICTURE

ב. שדות אלפבתיים או אלפנומריים

הכללים לרישום ערכים של שדה אלפנומרי הם:

- תוכן השדה יכול להכיל כל תו שהוא, מלבד גרש. בשדות אלפבתיים - רק אותיות, מקף ומרווח.
- התוכן המוגדר יוגבל בגרש מימין ומשמאל.
- אם מספר תוי התוכן המוגדר קטן ממספר התוים המגדירים את אורך השדה, השדה יושלם עם מרווחים מימין לתוכן המוגדר. אם השדה קטן ממספר תוי התוכן-יקוצצו תוים מימין.

דוגמאות:

```
77 SADE PIC X(4) VALUE 'AB·#'.
77  SADE  PIC X(10) VALUE '*'.
77 NAME PIC A(5) VALUE 'JØHN'.
```

שדה עם תוכן של תו אחיד

ניתן להגדיר תוכן התחלתי של שדה שכל התוים בו זהים, מבלי לחזור עליהם מספר פעמים כאורך השדה.

האפשרויות הן אלו:

א. VALUE SPACE. ימלא את כל השדה בתוכן התחלתי של מרווחים.

ב. VALUE ZERØ. תוכן התחלתי של אפסים.

ג. VALUE HIGH-VALUE. תוכן התחלתי של תוים עם הערך הבינרי הגבוה ביותר במערכת המחשב (בי.ב.מ./370 התו X'FF').

ד. VALUE LØW-VALUE. תוכן התחלתי של תוים עם אפסים בינריים (X'00').

ה. VALUE ALL 'יתו'. כשבין הגרשיים יצויין אותו תו שימלא את השדה המוגדר.

הערה: לשדות נומריים ניתן לציין VALUE ZERØ בלבד.

דוגמאות:

```
77 SADE1 PIC X(6) VALUE SPACE.
77 SADE2 PIC S9(8) VALUE ZERØ.
77 SADE3 PIC X(7) VALUE ALL '*'
```

במקרה האחרון נקבל שדה בן שבעה תוים עם ערך התחלתי של שבע כוכביות (*).

הגדרה מחדש לשטח זיכרון - REDEFINES

לעתים מתעורר הצורך להגדיר מחדש שטח זיכרון שהוגדר קודם לכן. (לדוגמה: סוגים שונים של רשומות במבנים שונים שנקראות לאותו שטח בזיכרון).

הגדרה מחודשת זו נעשית ע"י משפט REDEFINES, שנוסף להגדרה המחודשת כדי להדגיש שהגדרה זו אינה מקצה שטח נוסף בזיכרון.

מבנה ההגדרה:

(מספר רמה) (שדה 1) REDEFINES (שדה 2) [תאור שדה].

לדוגמה:

שדה מסוים ברשומה מכיל, לפי קוד מסוים המופיע ברשומה, שני ערכים שונים:
- מספר תעודת זהות של העובד
או - שם המחלקה בה הוא מועסק.

```
------
04 MSP-TZ PIC 9(7).
------
04 MACHLAKA REDEFINES MSP-TZ PIC X(7).
------
```

שדה באורך 7 המוגדר כשדה נומרי לצורך מספר תעודת הזהות ואותו שדה מוגדר כשדה אלפנומרי לצורך שם המחלקה.

אפשר להגדיר מחדש גם קבוצות של שדות, לדוגמה תאריך יכול להיות מוגדר: יום, חודש, שנה או: יום בשנה, שנה.

```
03 DATE.
   04 DAY PIC 99.
   04 MØNTH PIC 99.
   04 YEAR PIC 99.
03 NEW-DATE REDEFINES DATE.
   04 DAY-IN-YEAR PIC 999.
   04 YEAR        PIC  99.
   04 FILLER      PIC   X.
```

שים לב שבמקרה זה הוראת REDEFINES לא ניתנה לשדה אלמנטרי ולכן היא לא כללה את תאור השדה.

בכל מקרה ההגדרה החדשה אינה מקצה שטח זיכרון בהמשך להגדרה הקודמת אלא מגדירה מחדש אותו שטח זיכרון שהוגדר כבר. אפשר לבצע מספר הגדרות מחודשות בזו אחר זו.

כללי ההגדרה המחודשת של זיכרון הם:

- ההגדרה המחודשת תבוא מיד לאחר ההגדרה הראשונה.
- אסור להגדיר מחדש שטח המוגדר ברמה 01. (ראה בדוגמה הבאה כיצד מתגברים על מגבלה זו כשרוצים להגדיר מחדש רשומות שלמות: מגדירים רמה גבוהה יותר, פיקטי- בית, המיועדת רק לצורך זה).
- אורכו של השטח בהגדרה החדשה לא יהיה גדול מאורך השטח שהוגדר בהגדרה הראשונה.

תרגיל מסכם להגדרות נתונים

תאור הבעיה

הגדר שטח זיכרון שיכיל לסרוגין שתי רשומות במבנה שונה. ברשומה ב' יכו- לות להיות שתי אפשרויות לחלק מהשדות. הרשומות מתיחסות לפרטי חשבונות של לקו- חות:

רשומה א': סוג רשומה - 2 תוים
מספר לקוח - 6 תוים
כתובת לקוח - 20 תוים (רחוב - 8, מספר בית - 3, עיר - 9).

רשומה ב': סוג רשומה - 2 תוים
מספר לקוח - 6 תוים
שם לקוח - 10 תוים
סוג חשבון - 2 תוים
(*) מספר חשבון - 8 תוים

(*) מספר החשבון יהיה מספר אחד בן 8 ספרות או 2 מספרים נפרדים בני 4 ספרות כל אחד, לפי סוג החשבון.

פתרון התרגיל

תאור הרשומות:

א'

רשומה				
כתובת הלקוח			מספר	סוג
עיר	בית	רחוב	לקוח	רשומה

ב' אפשרות I

רשומה					
מספר חשבון	מספר חשבון	סוג חשבון	שם לקוח	מספר לקוח	סוג רשומה

ב' אפשרות II

רשומה				
מספר חשבון	סוג חשבון	שם לקוח	מספר לקוח	סוג רשומה

הגדרת הרשומות תהיה כדלקמן:

```
      |A  |B
   ---+---+----
      |01  RESHUMØT.
           02 RESHUMA-ALEF.                                     רשומה א
              04 SUG-A  PIC  XX.
              04 MISPAR-LAKUAH-A  PIC 9(6).
              04 KTØVET-A.
                 06 REHØV-A  PIC X(8).
                 06 MISPAR-BAIT-A PIC 9(3).
                 06 IR-A  PIC X(9).
           02 RESHUMA-BET REDEFINES RESHUMA-ALEF.               רשומה ב
              04 SUG-B  PIC XX.
              04 MISPAR-LAKUAH-B PIC 9(6).
              04 SHEM-B PIC X(10).
              04 SUG-HESHBØN-B PIC 99.
              04 MSP-HSH-B1.                                    ב-אפשרות I
                 06 M-H-1 PIC 9(4).
                 06 M-H-2 PIC 9(4).
              04 MSP-HES-B2 REDEFINES MS-HSH-B1  PIC 9(8).      ב-אפשרות II
```

<u>הערות:</u>

א) בדוגמה יש הגדרה מחדש (שורה 17) בתוך שטח שהוא בעצמו הגדרה מחדש של שטח זיכרון אחר (שורה 9). הדבר חוקי בהחלט ואין הגבלה למספר ההגדרות מחדש המותרות, כל עוד שומרים על כללי ההגדרה.

ב) בכל מקום בתכנית ניתן להתיחס לכל אחד מהשמות שהוגדרו בשתי הרשומות בלי קשר לרשומה נמצאת בזיכרון באותו רגע. כלומר, כל השדות "קיימים" ומוכרים והבחירה בין הרשומה הראשונה לבין השניה היא לוגית לאחר בחינת סוג הרשומה ע"י המתכנת.

ג) ברשומה ב' אפשר היה להמנע מלהגדיר את השדות סוג הרשומה ומספר הלקוח (שורות 10-11) כי אלה שדות שנמצאים <u>בדיוק</u> באותו מקום בזיכרון שבהם נמצאים סוג הרשומה ומספר הלקוח שברשומה א', וגם ניתן להתיחס אליהם באותו שם.
במקרה כזה אפשר היה לנהוג כך:
ברשומה ב' לציין FILLER באורך השוה לסכום האורכים של שני השדות הראשונים (8 תוים), ולהגדיר מחדש רק את החלק <u>השונה</u> של הרשומה, ולא את כולה כפי שנעשה.
כלומר:

```
02  RESHUMA-BET REDEFINES RESHUMA-ALEF.
    04 FILLER PIC X(8).
    04 SHEM-B PIC X(10).
    etc.
```

פרק 5: PROCEDURE DIVISION

תאור ומבנה

PROCEDURE DIVISION הינה החטיבה המרכזית של התוכנית המכילה את הוראות העיבוד של הנתונים, כלומר - את גוף התוכנית.

החטיבה מורכבת מקטעי תוכנית או סעיפים (Paragraphs), הקטעים - ממשפטים (Sentences), והמשפטים - מפסוקים (Statements).

החלוקה לקטעים הינה לוגית על פי צרכי כל תוכנית ותוכנית, והיא מאפשרת למתכנת לרכז קבוצות הוראות בקטע אחד ולהבדילן מקטעים אחרים של תוכניתו. לכל קטע יש שם-קטע (paragraph-name) אשר מאפשר להתיחס אליו בהוראות הפנייה או הסתעפות, כלומר להפסיק בצוע רצוף של הוראות לפי סדר כתיבתן בקטע אחד ולעבור לפי הצורך לבצוע הוראות בקטע תוכנית אחר. אפשר לראות את סדרת ההוראות הכלו-לה בקטע אחד כמו שגרה (סברוטינה) שאפשר לבצע אותה ע"י קריאה בשמה (שם-קטע).

בכל קטע ישנו משפט אחד או יותר ובכל משפט - פסוק אחד או יותר. כל משפט חייב להסתיים בנקודה הצמודה אל התו האחרון שלו. הפסוק הינו הוראת ביצוע אחת עם כל האופרנדים והמציינים שלה. זהו צרוף של מלים וסימנים המהווה יחידה לוגית אחת. לאחר הפסוק (בצמוד לתו האחרון שלו) יכולה לבוא הפרדה ע"י תו רווח, פסיק וגם נקודה. במקרה האחרון הפסוק הינו גם משפט.
דוגמה לקטע תוכנית:

```
PRØCEDURE DIVISIØN.
-------------------
DISTANCE. MULTIPLY HØURS BY SPEED GIVING RESULT,
          MØVE RESULT TØ PRINT-AREA.
---------------
```

בתוכנית ישנו קטע (DISTANCE) שבו מחושב המרחק על פי מכפלת זמן הנסיעה (HOURS) במהירות (SPEED). שדה התוצאה (RESULT) מועבר לשדה המתאים (PRINT-AREA) בשורה המיועדת להדפסה. בקטע ישנו משפט אחד המורכב משני פסוקים המופרדים בפסיק. בסוף הקטע (המשפט האחרון) ישנה נקודה.

כללי כתיבה

א. שם קטע

שם הקטע יכלול אותיות (A-Z), ספרות (0-9) ומקף. הוא יתחיל באות ולא יהיו בו מקפים רצופים. השם נכתב באזור א (כלומר החל מטור 8) ולאחריו נקודה, שאחריה מרווח אחד לפחות. ההוראה הראשונה של הקטע יכולה להכתב באותה שורה (אותו כרטיס) של השם או בשורה שאחריו.

ב. משפטים ופסוקים - הוראות קובול

משפטי קובול נכתבים באזור ב (מטור 12 ואילך) במבנה חפשי תוך שמירת הכלל שבין מלה למלה יפריד לפחות מרווח אחד. המבנה החפשי מתבטא במספר חפשי של מרווחים בין המלים, במעבר חפשי (ללא כל סימון מיוחד) משורה לשורה ואף באפשרות לכתוב כל מלה בשורה נפרדת. כלומר, למתכנת יש אפשרות לכתוב את התוכנית בצורת עריכה שתקל עליו את עבודתו. ניתן להשתמש בפסיק (,) להפרדה בין פסוק אחד למשנהו בתוך המשפט. הפסיק נועד לנוחות הקריאה ואין לו כל תפקיד לוגי. הפסיק יכתב צמוד למלה שלפניו ולאחריו - תו רווח אחד לפחות.

לאחר המשפט האחרון בקטע, כלומר לפני "שם-קטע" חדש, חייבת לבוא נקודה. נקודה חייבת להיות גם בסיום משפטי התנייה (IF). אם לא רושמים נקודה בגוף הקטע, הופכות כל ההוראות לפסוקים של משפט אחד. הנקודה נרשמת, כמו הפסיק צמוד למלה שלפניה ועם תו רווח אחד לפחות אחריה.

תחילת וסיום התוכנית

אם נצמצם את הגדרת התוכנית ל"סדרת הוראות המיועדות לבצע פעולות עיבוד מוגדרות" נוכל לכנות את PROCEDURE DIVISION כ"חטיבת תוכנית העיבוד".

א. תחילת התוכנית

כאשר מערכת ההפעלה מעבירה את הפיקוח אל התוכנית והיא נכנסת לפעולה, מתבצעת בה ההוראה הראשונה הרשומה בקטע הראשון בחטיבה זו. כאן יש להעיר שלמעשה תתבצענה הוראות רבות לפני הוראה זו, אולם הן "שקופות" למשתמש. הוראות אלו כלולות במבוא של כל תוכנית קובול והן קשורות לניהול התוכנית ע"י השפה

ואין למשתמש שליטה בהן. המהדיר של קובול משלב קטעי תכנות שונים במהלך תוכנית המשתמש המיועדים לניהול התכנות ולביצוע פעולות אשר המשתמש בשפה עילית פטור מהן. פרטים על כך אפשר לקרוא בספרות המתאימה לשפת קובול, במיוחד בחוברות העוסקות בתאור המבנה הלוגי של התוכנית מבחינתו של המהדיר.

ב. סיום התוכנית

בניגוד להוראת הפתיחה שאינה מיוחדת אלא זו הכתובה ראשונה, יש הוראה מיוחדת לסיום התוכנית ולהעברת הפיקוח חזרה אל מערכת ההפעלה:

STØP RUN

הוראה זו היא ההוראה האחרונה שתתבצע, אך אינה בהכרח ההוראה האחרונה הכתובה בתוכנית. הוראה זו יכולה להופיע יותר מפעם אחת בתוכנית.

```
    A  B
1   PRØCEDURE DIVISIØN.
2   NEW. ACCEPT A CØDE FRØM CØNSØLE.
3        IF CØDE = 1 GØ TØ CHECK.
4   NEXT. -----------
5         -----------
6        STØP RUN.
7   CHECK. ADD A TØ CØDE.
8         -----------
9        GØ TØ NEXT.
```

הסבר: ההוראה הראשונה שתתבצע בתוכנית כתובה בשורה 2, והיא תבצע קריאת נתון ממכונת הכתיבה של המפעיל ותעביר אותו לשדה CODE. אחר כך תתבצע בדיקה אם CODE=1. אם כן - יבוצע סיעוף אל קטע התוכנית CHECK (בשורה 7) ומשם (משורה 9) יבוצע סיעוף אל קטע התוכנית NEXT (בשורה 4). בסיום קטע התוכנית הזה ישנה הוראת סיום התוכנית (בשורה 6). אם התשובה במשפט ההתנייה (בשורה 3) היא שלילית, תמשיך התוכנית בקטע NEXT (בשורה 4).

הוראות קלט ופלט בסיסיות

בשלב זה של התכנות בקובול נשתמש באמצעי קלט פלט (מכונת הכתיבה של מפעיל המחשב, קוראת ומנקבת כרטיסים ומדפסת) ובשיטות בהזנה והפקה של נתונים ללא צורך בהגדרה מפורשת. הוראות הקלט והפלט בעצמן מגדירות את היחידה ואת הפעולה הנדרשת.

שיטה זו טובה למקרים בהם כמויות הנתונים קטנות כגון הזנת קודים לתוכנית, הזנת תאריך, הודעה מהתוכנית למפעיל, דווח על שגיאות או על חריגים, התרעות ועוד. בנפח גדול של נתונים נשתמש בהגדרת קבצים ביחידות קלט ופלט מגנטיות כגון סרטים ודיסקים. בדיסקים שיטות אירגון הנתונים יכולות להיות שונות מהשיטה הסודרת (Sequential) ובכל המקרים - ניהול ופיקוח על פעולות קלט ופלט כוללים אפ- שרויות רבות יותר וחלקם נעשה ע"י מערכת ההפעלה.

<u>הוראות קלט - ACCEPT</u>

הוראה זו משמשת לקבלת נתונים מהמפעיל בעת ביצוע התוכנית, באמצעות מכונת הכתיבה או הצג שלרשותו, או לקריאת כרטיסים מנוקבים. אין צורך להגדיר קבצים. להוראה שתי צורות:

א. <u>קריאת כרטיסים מנוקבים:</u> ACCEPT (שם שדה).

כל ביצוע של ההוראה גורם לקריאת כרטיס אחד בקוראת הכרטיסים. תוכנו של הכרטיס מועבר לשדה (רשומה) בן 80 תוים ששמו צויין בהוראה והוגדר ב-DATA DIVISION. המתכנת חייב לקבוע בעצמו את מבנה הכרטיס האחרון בקובץ ולבדוק לאחר כל קריאה אם כרטיס זה התקבל, ואז להפסיק את הקריאה (זוהי בדיקה ל"כרטיס אחרון").

ב. <u>קריאת נתונים מעמדת המפעיל:</u> ACCEPT (שם שדה) FROM CONSOLE.

המפעיל מקבל איתות שעליו לספק נתונים מעמדת הפיקוח שלו. התוים שיוזנו על ידו באמצעות לוח המקשים יועברו לשדה המצויין בהוראה. מספר התוים שהמפעיל ידרש לתקתק יקבע על פי אורך השדה כפי שהוגדר ב-DATA DIVISION.

הוראות פלט - DISPLAY

להוראה זו שלש צורות בהתאם לאמצעי הפלט שאליו מכוונים את הנתונים.

א. DISPLAY (שם שדה)

תוכנו של השדה (שורה) יודפס במדפסת. ביצוע כל הוראה DISPLAY גורם להדפסת שורה אחת במדפסת. אין כל אפשרות לפקח על ההדפסה, כמו דילוג לדף חדש להדפסת כותרות וכ"ד, שהן פעולות ההכרחיות בהדפסת קבצי דוחות.

ב. DISPLAY (שם שדה) UPON CONSOLE

תוכן השדה יודפס במכשיר הפקוח של מפעיל המחשב.

ג. DISPLAY (שם שדה) UPON SYSPUNCH

תוכן השדה ינוקב בכרטיס מנוקב.

תרגיל

יש לקרוא כרטיס מקוראת הכרטיסים, להוסיף לו תאריך הביצוע ולהדפיסו. התאריך יסופק על ידי מפעיל המחשב. מבנה הפלט הוא:

PRINT DATE
(תוכן הכרטיס 80 טורים) ___ XX/XX/XX
תאריך 3 רווחים

התוכנית:

```
A   B
IDENTIFICATIØN DIVISIØN.
PRØGRAM-ID. 'TARGIL'.
DATA DIVISIØN.
WØRKING-STØRAGE SECTIØN.
77  MESSAGE PIC X(10) VALUE 'PRINT DATE'.
01  KARTIS-DATE.
    03 KARTIS PIC X(80).
    03 FILLER PIC X(3) VALUE SPACE.
    03 DATE PIC X(8).
PRØCEDURE DIVISIØN.
KETA-1.
    ACCEPT KARTIS.
    DISPLAY MESSAGE UPØN CØNSØLE.
    ACCEPT DATE FRØM CØNSØLE.
    DISPLAY KARTIS-DATE.
    STØP-RUN.
```

הסבר:

א. שים לב למבנה התוכנית הכולל:
- סדר החטיבות - DIVISIONS
- כתיבה באזורים A, B
- סימני פיסוק והפרדה

ב. בשורה 5 מוגדר שדה אלמנטרי בודד בקוד 77, אשר מכיל ערך התחלתי המיועד לשמש ככותרת שורת הנתונים שתודפס במהלך התוכנית.

ג. בשורה 6 מופיעה הגדרה של נתון קבוצתי הכוללת את כל השדות המיועדים להדפסה. הנתונים האלמנטריים שלו הם KARTIS ו-DATE. אל שני שדות אלה נקרא את הנתו- נים כמפורט בהמשך.

ד. בשורה 8 מוגדר שדה של 3 תוי רווח להפרדה בין שני שדות הנתונים בשורת ההדפסה.

ה. בשורה 10 מוגדרת חטיבת התוכנית המכילה את הוראות הביצוע ובה סעיף אחד ששמו ניתן בשורה 11.

ו. מהלך התוכנית כולל: קריאת כרטיס מנוקב לשדה KARTIS (שורה 12), הדפסת השדה MESSAGE במכונת הכתיבה של המפעיל (שורה 13), קריאת שדה תאריך שיוזן ע"י המפעיל במכונת הכתיבה (שורה 14) ולבסוף-הדפסת השדה KARTIS-DATE כולו (שורה 15) המכיל את השדות KARTIS ו-DATE.

ז. בשורה 16 ישנה ההוראה האחרונה בתוכנית והיא גם ההוראה האחרונה שתתבצע. משמעותה - הפסקת ביצוע התוכנית והעברת הפיקוח על מערכת ההפעלה.

הוראת העברת נתונים - MOVE

המבנה של ההוראה:

MØVE (שם שדה 1) TØ (שם שדה 2)

ההוראה מעבירה את תוכן שדה 1 לשדה 2. תוכן שדה 1 נשאר כפי שהיה לפני ביצוע ההוראה, ותוכן שדה 2 מכיל התוכן החדש שהועבר אליו. יש לזכור שאין זו פעולת "העברה" ובמובן המקובל אלא פעולת העתקה, מכיון שהנתון ה"מועבר" נשאר במקום שהיה בו לפני הפעולה.

כללי העברת נתונים

א. העברות מותרות

אל שדה שהוגדר כאלפנומרי מותר להעביר נתונים משדה כלשהו (מספרי, אלפבתי או אלפנומרי).

אל שדה אלפבתי מותר להעביר נתונים משדה אלפבתי בלבד, ובהקבלה - אל שדה מספרי מותר להעביר נתונים משדה מספרי בלבד.

ב. קצוץ והארכה:

א. שדות אלפנומרים ואלפבתים: אם השדה המקבל ארוך מהשדה המוסר, התוים הנותרים מימין לנתונים שהועברו, ימולאו במרווחים.
- אם השדה המקבל קצר מהשדה המוסר, יקוצצו התוים מימין מכיון שההעברה נעשית לפי תוים משמאל לימין.

ב. שדות מספריים - אם השדה המקבל קצר מהשדה המוסר, יקוצצו הספרות המש- מעותיות כי העברת הספרות היא מימין לשמאל (מלבד הספרות שמימין לנקו- דה העשרונית בהן ההעברה היא משמאל לימין).
- אם השדה המקבל ארוך מהשדה המוסר, יתווספו אפסים משמאל לספרות השלמים ומימין לספרות הדצימליים כדי לשמור על ערכו האריתמטי של המספר.

בהעברת שדות מספריים נעשית התאמה מלאה של מיקום הנקודה העשרונית של השדה המוסר והשדה המקבל כדי לשמור על ערכו הנכון של המספר. אם מספר הספרות מימין ומשמאל לנקודה העשרונית שונה בשני השדות - יקוצצו או יתווספו ספרות כפי שהוסבר קודם.

העברה מרובה

אם רוצים להעביר תוכן שדה אחד למספר שדות ניתן להשתמש באופציה המורחבת:

MOVE (שם שדה1) TO (שם שדה2) (שם שדה3) ....

תוכן שדה 1 יועבר לכל השדות שיצויינו במשפט. לדוגמה, העברה של TOTAL ל-3 שדות:

```
MØVE TØTAL TØ LINE-1 LINE-2 SIKUM.
```

דוגמאות:

ההוראה היא MØVE ALPHA TØ BETA

	הגדרת ALPHA	הגדרת BETA	תוכן ALPHA	תוכן BETA לאחר ההעברה
1	S9(4)	S9(7)	1234	0001234
2	S999	S99	789	89
3	S999	S999V99	432	43200
4	X(3)	X(6)	ABC	ABCbbb
5	X(4)	XX	ABCD	AB

הסבר:

1 - שדה BETA גדול יותר ולכן התווספו אפסים משמאל למספר שהועבר אליו.
2 - שדה BETA קטן יותר ולכן קוצצה הספרה השמאלית של המספר שהועבר.
3 - המספר הועבר בהתחשב במיקום הנקודה העשרונית, כלומר - שמירת ערכו 432.00.
4 - שדה אלפביתי הועבר לשדה גדול יותר ולכן התווספו מרווחים מימין.
5 - שדה אלפביתי, הועבר לשדה קטן יותר ולכן נקצצו תוים מצד ימין.

תוראות לביצוע פעולות חשבוניות

הוראות קובול הבסיסיות מאפשרות ביצוע 4 פעולות החשבון. כל פעולה תרשם כפסוק נפרד ותפיק שדה תוצאה אחד. אפשר גם להשתמש בהוראה מורכבת COMPUTE לביצוע סדרה של פעולות לשם קבלת תוצאה אחת לכולן.

חיבור - ADD

להוראת החיבור צורות אחדות.

א. הוראת חיבור של שני מספרים:

ADD (שם שדה1) TØ (שם שדה2)

תוכן השדה הראשון יחובר לתוכן השדה השני. התוצאה מתקבלת בשדה השני (במקום הערך שהיה בו קודם).

דוגמה: ADD AA TØ BB.

	שדה BB	שדה AA
לפני ביצוע ההוראה	5	3
לאחר ביצוע ההוראה	8	3

כבשאר ההוראות המבצעות פעולות חשבוניות, שני השדות חייבים להיות מספריים, הן בהגדרתם והן בתוכנם, בעת ביצוע ההוראה.

ב. הוראת חיבור של מחוברים אחדים:

ADD (שם שדה1) (שם שדה2) (שם שדה3) ... GIVING (שם שדהת)

בצורתה המורחבת מחברת ההוראה תוכן כל השדות שמשמאל למלת המפתח ומעבירה את התוצאה אל השדה שצויין מימין למלה GIVING (שם שדה n). שדה זה אינו משתתף בפעולה ותוכנו הקודם נמחק כאשר נכתבת בו תוצאת הסיכום.
לדוגמה:

ADD AA BB CC DD GIVING EE

משמעות ההוראה: EE = AA + BB + CC + DD

סיכום השדות מועבר לשדה EE.

ג. הוראת חיבור עם התניית מצב שגיאה:

{ADD ... TØ ..... / ADD ... GIVING....} ØN SIZE ERRØR (פסוק קובול)

אפשר להשתמש בהתניית מצב שגיאה בכל אחת משתי הצורות הקודמות של הוראת ADD. התוספת ON SIZE ERROR מאפשרת לגלות אם היה אובדן ספרות משמעותיות (גלישה - overflow) בעת ביצוע הפעולה, כלומר אם התוצאה שנתקבלה גדולה מאורך השדה המקבל אותה.

אם היתה גלישה יתבצע פסוק הקובול שנכתב בהמשך למלת המפתח ON SIZE ERROR. הוא יכול להורות על ביצוע חישוב אחר, הדפסת הודעה מתאימה או אף הפסקת הביצוע של התוכנית.

אם לא היתה גלישה, ביצוע התוכנית יימשך החל מהמשפט הבא אחרי הנקודה המסיימת את פסוק הקובול שצויין בהמשך ל-ON SIZE ERROR. שים לב, הוראת חיבור עם התניית שגיאה היא משפט המחייב נקודה בסופו.

<u>דוגמה:</u>

```
1  ADD AA TØ BB ØN SIZE ERRØR MØVE ZERØ TØ BB.
2  MØVE CC TØ DD.
```

אם נוצר מצב של גלישה בעת ביצוע הוראה (1) השדה BB יאופס כתוצאה מהפסוק הרשום בהמשך ל-ON SIZE ERROR. לאחר מכן תמשיך התוכנית בביצוע המשפט בשורה 2. אם <u>לא</u> היתה גלישה לאחר החיבור, התוכנית תמשיך להתבצע החל מהמשפט <u>שאחרי</u> הנקודה, כלומר המשפט בשורה 2.

משמעות קטע תוכנית זה הוא שכאשר יש גלישה בשדה BB - מאפסים אותו וממשיכים.

שים לב!

אם לא משתמשים באופציה ON SIZE ERROR אי אפשר לגלות מצב של אובדן ספרות משמעותיות.

ד. <u>הוראת חיבור עם עיגול התוצאה:</u>

ADD .... <u>RØUNDED</u> [ØN SIZE ERRØR...]

מלת המפתח ROUNDED גורמת <u>לעיגול</u> התוצאה כשהשדות המעורבים מוגדרים עם ספרות עשרוניות. עלול להיות קיצוץ של ספרות שמימין לנקודה העשרונית כאשר הגדרת השדה המקבל את התוצאה מכילה פחות ספרות עשרוניות במכוון. למשל בשדה של ערך כספי אין צורך ביותר משתי ספרות עשרוניות.

גם בהוראה זו אפשר להוסיף את התניית מצב השגיאה של גלישת נתונים.
לדוגמה נתונים השדות הבאים:

```
77 AA PIC S9(4)V99 VALUE 100.43.
77 BB PIC S9(3)V99 VALUE 80.14.
77 CC PIC S9(4)V9.
```

א. לאחר הפעולה ADD AA BB GIVING CC.
התוצאה האמיתית תהיה 180.57 אך מאחר ושדה CC מוגדר עם ספרה אחת בלבד מימין לנקודה העשרונית, יתבצע <u>קיצוץ</u> ונקבל: 180.5.

ב. לאחר הפעולה ADD AA BB GIVING CC RØUNDED.
במקום קיצוץ יתבצע <u>עיגול</u> על ידי הוספת 5 בספרה הימנית ביותר "המקוצצת" והתוצאה תהיה 180.6.

דוגמה נוספת:

```
77 AA PIC S9(3)V99 VALUE 100.42.
77 BB PIC S9(3)V99 VALUE 80.11.
77 CC PIC S9(3)V9.
```

הוראות הביצוע הן כמו בדוגמה הקודמת, אולם כאן התוצאה האמיתית היא 180.53. אם נוסיף 5 לספרה הימנית ביותר, הנקצצת, נקבל 180.58. מכאן שבשני המקרים, בקיצוץ ללא עיגול או בעיגול, נקבל את התוצאה 180.5.

חיסור - SUBTRACT

להוראת החיסור צורות אחדות.

א. הוראת חיסור של שדה אחד משדה שני:

SUBTRACT (שם שדה1) FRØM (שם שדה2)

תוכן שדה 1 יופחת משדה 2 והתוצאה תתקבל בשדה 2.

ב. הוראת חיסור עם תוצאה בשדה אחר:

SUBTRACT (שם שדה1) FRØM (שם שדה2) GIVING (שם שדה3)

תוצאת ההפחתה של שדה 1 משדה 2 תתקבל בשדה 3. בשדה 1 ובשדה 2 ישמרו הערכים הקודמים.

ג. הוראת חיסור עם התניית מצב שגיאה ועם עיגול התוצאה:

בשני מקרים אלה ניתן להשתמש באופציה ON SIZE ERROR ו-ROUNDED כמו בהוראות החיבור.

דוגמאות:

```
SUBTRACT ALEF FRØM BET RØUNDED.
SUBTRACT ALEF FRØM BET GIVING GIMEL.
SUBTRACT ALEF FRØM BET ØN SIZE ERRØR MØVE 1 TØ SADE-INDICATØR.
```

שים לב להבדיל בין שתי האפשרויות הבאות:

```
a)  SUBTRACT ALEF FRØM BET. MØVE BET TØ GIMEL.
b)  SUBTRACT ALEF FRØM BET GIVING GIMEL.
```

בשני המקרים נקבל ב-GIMEL את תוצאת החיסור: BET פחות ALEF. במקרה (a) התוצאה תהיה <u>גם</u> ב-BET בעוד שבמקרה (b) תוכנו של BET נשמר ויישאר כפי שהיה לפני ביצוע ההוראה.

<u>כ פ ל - MULTIPLY</u>

להוראת הכפל צורות אחדות.

א. <u>הוראת כפל של שני גורמים</u>

MULTIPLY (שם שדה1) BY (שם שדה2)

מכפלת שדה 1 בשדה 2. תוצאת הכפל מתקבלת בשדה השני.

ב. <u>הוראת כפל כאשר התוצאה מתקבלת בשדה אחר</u>

MULTIPLY (שם שדה1) BY (שם שדה2) GIVING (שם שדה3)

תוצאת המכפלה של שדה 1 בשדה 2 תתקבל בשדה 3. שדה 1 ושדה 2 ישארו ללא שינוי.

ג. גם בהוראות הכפל ניתן להשתמש באופציות ROUNDED ו- ON SIZE ERROR כמו בהוראות חיבור וחיסור.

<u>דוגמה:</u>

```
MULTIPLY AA BY AA GIVING BB RØUNDED
         ØN SIZE ERRØR MØVE ZERØ TØ BB.
```

ההוראה הנ"ל מעלה את המספר שבשדה AA ברבוע (AA כפול AA) ומעבירה את התוצאה מעוגלת לשדה BB. אם שדה זה קטן מלהכיל את התוצאה (כלומר ספרות משמעותיות אבדו), יאופס השדה BB.

חילוק - DIVIDE

א. הוראת חילוק של שני מספרים:

DIVIDE (שם שדה1) INTØ (שם שדה2)

שדה 2 מחולק בשדה 1 והתוצאה מתקבלת בשדה 2.

ב. הוראת חילוק עם תוצאה בשדה אחר:

DIVIDE (שם שדה1) INTØ (שם שדה2) GIVING (שם שדה3)

תוצאת החילוק תתקבל בשדה 3.

ג. ניתן להשתמש באופציות ROUNDED ו-ON SIZE ERROR.
SIZE ERROR במקרה של חילוק יתקבל כאשר מבוצע חילוק באפס.

ד. בהוראת החילוק אפשר לשמור את השארית. לשם כך אפשר להוסיף בהוראה:
... REMAINDER (שם שדה4)

דוגמאות:

$\frac{\text{DISTANCE}}{\text{TIME}}$ = SPEED → DIVIDE TIME INTØ DISTANCE GIVING SPEED RØUNDED.

$\frac{\text{SUM}}{\text{FACTØR}}$ = RESULT → DIVIDE FACTØR INTØ SUM GIVING RESULT.

פעולות חשבוניות מרובות וההוראה - COMPUTE

ההוראה COMPUTE מאפשרת לבצע סדרת פעולות חשבוניות במשפט אחד ולקבל את התוצאה בשדה אחד המוגדר בהוראה. מבנה ההוראה:

CØMPUTE (שדה 1) [RØUNDED] = (ביטוי חשבוני) [ØN SIZE ERRØR...].

תוצאת הביטוי החשבוני מתקבלת תמיד בשדה 1 אשר רשום ליד ההוראה COMPUTE. הפעולה החשבונית בין השדות מצויינת ע"י אחד מהאופרטורים החשבוניים ולא ע"י מלים כמו בהוראות החשבוניות הבסיסיות. משני צידי האופרטור חייב להיות תו

רווח אחד לפחות. האופרטורים המותרים:

חיבור +	כפל *	העלאה בחזקה **
חיסור -	חילוק /	(שתי כוכביות צמודות)

סדר ביצוע הפעולות לפי הכללים המקובלים באלגברה:

- העלאה בחזקה מתבצעת ראשונה,
- חילוק וכפל מתבצעים לאחר מכן,
- חיבור וחיסור מתבצעים לאחרונה.

מקרה פרטי של ההוראה COMPUTE הוא כאשר במקום הביטוי החשבוני נרשם שם של שדה אחד בלבד. פעולה זו מקבילה לפעולת ההוראה MOVE. אך יש לזכור שבהוראה COMPUTE מותר להעביר שדה מספרי <u>בלבד.</u>

למשל: .CØMPUTE SPEED = FIELDA

פרושו ששדה SPEED מקבל את ערכו של FIELDA.

דוגמאות:

א. CØMPUTE AA = BB + CC - DD / EE

שים לב לרווחים שמשני צידי האופרטורים.

סדר הפעולות: חילוק, חיבור או חיסור, ולאחר מכן העברה לשדה AA.

ב. CØMPUTE AA = BB + CC * DD

בתחילה תתבצע המכפלה CC * DD, אליה יחובר השדה BB ולאחר מכן תועבר התוצאה לשדה AA.

<u>סוגריים</u>

אפשר לשנות את סדר ביצוע הפעולות החשבוניות ע"י שימוש בסוגריים, כמקובל באלגברה. אין הגבלה למספר הסוגריים אך נעשית בדיקה אם יש מספר שוה של סוגריים ימניים ושמאליים. הפעולה שבסוגריים הפנימיים ביותר תתבצע לראשונה, ולאחר מכן - זו שבסוגריים החיצוניים יותר, וכן הלאה.

דוגמאות:

א. CØMPUTE AA = (BB + CC) * DD

בתחילה יתבצע החיבור ואחר כך הכפל.

ב. COMPUTE M = ((A + B) * C) ** 2

סדר הפעולות: חיבור, כפל, העלאה בחזקה.

ג. COMPUTE A = (X - (B * Y)) / D

סדר הפעולות: כפל, חיסור, חילוק.

הסוגריים יכולים להיות צמודים לשם-השדה או צמודים זה לזה, אולם צריך להיות תו רווח אחד (לפחות) בינם לבין האופרטור החשבוני.

תרגיל לפעולות חשבוניות

נתונים השדות המספריים ALEF, BET, GIMEL, DALET.

יש לחשב שדה HEY לפי הנוסחה:

$$HEY = \frac{(ALEF + BET - GIMEL) * DALET}{GIMEL + BET}$$

התוצאה ב-HEY דרושה עם עיגול. במקרה של גלישת התוצאה יש לאפס את השדה HEY.

א. תכנות התרגיל

```
A   B
WØRKING-STØRAGE SECTIØN.
77  ALEF  PIC  S9(6)V99.
77  BET   PIC  S9(5)V99.
77  GIMEL PIC  S9(4)V99.
77  DALET PIC  S9(5).
77  HEY   PIC  S9(5)V99.
PRØCEDURE DIVISIØN.
    ADD ALEF TØ BET GIVING HEY RØUNDED.
    SUBTRACT GIMEL FRØM HEY RØUNDED.
    MULTIPLY DALET BY HEY RØUNDED ØN SIZE
       ERRØR MØVE ZERØ TØ HEY.
       ADD GIMEL TØ BET.
       DIVIDE BET INTØ HEY RØUNDED.
```

<u>הסבר:</u>

משתמשים בשדה התוצאה הסופי HEY לביצוע כל פעולות הביניים. לאחר חישוב הערך שבסוגריים (שורות 9-8) כופלים אותו בשדה DALET (שורות 10-11). אם יש "שגיאת גודל" (SIZE ERROR) מאפסים את השדה HEY וממשיכים.
מחשבים את הסכום במונה (שורה 12) ו"הורסים" את השדה BET אשר אין זקוקים לו יותר. לאחר מכן מחלקים את הערך של המונה מ-HEY במכנה הנמצא ב-BET (שורה 13). תוצאת החילוק תהיה בשדה HEY, מעוגלת.

ב. חישוב השדה HEY אשר נערך קודם לכן בשלבים, יבוצע במשפט אחד באמצעות ההוראה COMPUTE.

```
COMPUTE HEY ROUNDED =
     ((ALEF + BET - GIMEL) * DALET / (GIMEL + BET)
     ON SIZE ERROR MOVE ZERO TO HEY.
```

ליטרלים (literals) בהוראות

ליטרלים הם קבועים (constants) ומשתמשים בהם במשפטי קובול ב-PROCEDURE DIVISION בלי שהוגדרו קודם לכן ב-WORKING STORAGE עם ערך התחלתי (VALUE). הופעת הקבוע בהוראות גורמת להגדרתו האוטומטית ע"י המהדיר.

הליטרלים יכולים להופיע במקום "שם-שדה", אך <u>אינם יכולים</u> להופיע במקום "שם-שדה" <u>שמקבל</u> את התוצאה של הפעולה.

הליטרלים הם משני סוגים: מספריים ואלפנומריים. חוקי כתיבתם זהים לחוקי כתיבת ה-VALUE במשפטי הגדרת שדות (PICTURE). כאשר יש מספר עם סימן חיובי אפשר להשמיט את הסימן, אך סימן שלילי יש לכתוב תמיד כשהוא צמוד למספר, משמאלו.

<u>דוגמאות:</u>

```
MOVE 1 TO AA.
MOVE +1 TO CC.          (הסימן + צמוד לסיפרה)
MOVE 'KOTERET' TO SADEC.        (אלפנומרי)

ADD 1 TO MONE.
```

```
DIVIDE 5 INTØ 25 GIVING SADE.
SUBTRACT 1.5 FRØM MØNE RØUNDED.
CØMPUTE A = 7 * B / 2.5.
```

<u>אך אסור</u>:

```
MØVE SADE TØ 'KØTERET .
ADD 3 TØ 7.
```

במקרים אלו מופיע ליטרל במקום שדה שמקבל את התוצאה, ולכן <u>אסור</u> הדבר.

ליטרלים יכולים להופיע גם כקבועים בעלי תו אחיד המוגדרים ע"י המלים הבאות:

```
ZERØ    HIGH-VALUE   ALL 'תו'
SPACE   LØW-VALUE
```

<u>לדוגמה</u>:

```
MØVE ZERØ TØ MØNE.
MØVE SPACE TØ SHURA.
MOVE ALL '*' TØ SADE-CØHAVIØT.
```

הוראות הסתעפות

הוראות התוכנית מתבצעות בצורה רציפה בזו אחר זו על פי סדר כתיבתן בתוכ-ית. הוראות הסתעפות משמשות להפסקת הרצף ולמעבר לקטע אחר בתוכנית. הסיעוף מתבצע תמיד אל ראש הקטע ("שם-הקטע").

<u>הסתעפות לכתובות קבועות - GØ TØ</u>

א. <u>הסתעפות רגילה</u> (לא מותנית):

GØ TØ (שם קטע).

ההוראה מעבירה את פיקוח ביצוע התוכנית לקטע (paragraph) ששמו צויין בהוראה. ההוראה הבאה שתתבצע תהיה ההוראה הראשונה בקטע שאליו מתבצע הסיעוף.

ב. <u>הסתעפות מותנית בערך שדה</u>

GØ TØ (קטע 1) (קטע 2) (קטע 3) ... DEPENDING ØN (שם שדה).

הוראה זו הינה הסתעפות מותנית על פי תוכן שדה נתון ("שם-שדה"). השדה חייב להיות <u>מספרי</u> ולפי ערכו יקבע הקטע הסידורי ברשימה שאליו יתבצע הסיעוף.
אם הערך = 1 יבוצע סיעוף לקטע הראשון שברשימה, אם הערך = 2 יבוצע הסיעוף לקטע השני ברשימה, וכך הלאה. אם ערכו של השדה קטן מ-1 (אפס או שלילי) או גדול ממספר שמות הקטעים שברשימה - לא תתבצע הסתעפות וההוראה הבאה שתתבצע תהיה זו שרשומה לאחר הוראת ההסתעפות.

<u>דוגמאות:</u>

(1

```
   A      B
1  KETA.
2         ADD MØNE TØ MØNE2.
3         MØVE AA TØ BB.
4         GØ TØ KETA2.
5  KETA1.
6         MØVE ZERØ TØ MØNE2.  GØ TØ KETA.
7  KETA2.  ADD 1 TØ MØNE2 ØN SIZE ERRØR.
8         GØ TØ KETA1.
```

<u>הסבר:</u>

יש לבצע את סדרת ההוראות הכלולות ב-KETA ולאחריו להסתעף ל-KETA2 (שורה 4). עם סיומו של KETA2 מסתעפים ל-KETA1 (שורה 8). מבצעים קטע זה ומסתעפים ל- KETA (שורה 6). וכן הלאה, כלומר זוהי לולאה אינסופית שבאה להציג סיעוף בין שלשה קטעים.

(2

```
      MØVE -1 TØ MØNE.
  LØØP.
      ADD 1 TØ MØNE.
      GØ TØ AA BB DEPENDING ØN MØNE.
      GØ TØ LØØP.
  AA. -----

  BB. -----
```

<u>הסבר:</u>

דוגמה זו מיועדת להציג סיעוף המותנה בערך של שדה מסוים. בפעם הראשונה

MONE = 0 (שים לב לשורות 3,1) ולכן לא תתבצע ההסתעפות המותנית (שורה 4), אלא ההוראה הבאה שלאחריה (שורה 5) שהיא הוראת חזרה לראש הקטע - LOOP. בפעם השניה MONE = 1 ולכן תתבצע הסתעפות לקטע AA.

<u>שינוי כתובת בהוראת הסתעפות</u>

בדרך כלל כתובת ההסתעפות קבועה בהוראה GO TO. ישנם מצבים בהם יש צורך לשנות את כתובת ההסתעפות של ההוראה באופן דינמי, במהלך ביצוע התוכנית. פעולה זו מתבצעת ע"י ההוראה ALTER.

ALTER (שם קטע 1) TØ PRØCEED TØ (שם קטע 2).

"שם קטע 1" - שם סעיף שיש בו הוראה ... GO TO <u>יחידה ובלתי-מותנית</u>.
"שם קטע 2" - שם סעיף כלשהוא בתוכנית. לאחר ביצוע ההוראה ALTER תשתנה הכתובת בהוראה GO TO (הנמצאת ב"שם קטע 1") אל:

(שם קטע 1) GØ TØ (שם קטע 2).

<u>דוגמה:</u>

```
    A   B
1   AA. GØ TØ CC.
2   CC. MØVE ZERØ TØ MØNE.
3       ALTER AA TØ PRØCEED TØ BB.
4   BB. ------
```

<u>הסבר:</u>

כאשר התוכנית תגיע בפעם הראשונה ל-AA, תתבצע הסתעפות ל-CC. שם תבוצע הוראת MOVE בשורה 2. בסוף הקטע תוחלף כתובת ההסתעפות של AA ע"י הוראת ALTER: הוראת הסיעוף לקטע CC שהיתה במקור תוחלף להוראת סיעוף ל-BB. בפעמים הבאות כשהתוכנית תגיע ל-AA תבוצע הסתעפות לקטע BB, אלא אם כתובת ההסתעפות תוחלף שוב ע"י הוראה ALTER במקום אחר בתוכנית.

השימוש בהוראת ALTER מיועד בעיקר למטרות הבאות:

א. בתור מתג (switch). עפ"י תנאים מסויימים מפנים את ההסתעפות למקום זה או אחר בתוכנית.

ב. לביצוע פעילות חד-פעמית בתוכנית (ראה בדוגמה לעיל).
לאחר ביצוע קטע מסויים מחליפים את הוראת הסיעוף כדי שלא לחזור עליו פעם נוספת.

ג. השימוש בשגרות (סברוטינות) סגורות, ל"דריכת" כתובת החזרה של השגרה. במקרה זה השגרה תסתיים בקטע שכל הוראתו GO TO. לפני כל הסתעפות לשגרה תופעל הו-ראה לשינוי כתובת ההסתעפות הזו כדי לשוב לקטע הרצוי.

תרגיל להוראות הסתעפות

יש לקרוא קובץ כרטיסים. בכל כרטיס מנוקבים השדות IND ו-ALEF. יש לחבר את תוכן השדה ALEF אל המונים MONE1, MONE2, MONE3 על פי הערך של השדה IND (1, 2 או 3) בהתאמה.

אם תוכן שדה IND שונה מ-1, 2 או 3 יש לחבר את השדה ALEF אל MONE4.

פתרון התרגיל ראה בטופס הקידוד המצורף.

הסבר: בפתרון זה כלולה דוגמת תוכנית. שים לב לעובדה ששמות חטיבות ללא פרוט חייבים להופיע (שורה 1-05).

בצד שמאל של הטופס נרשמו בעמודות המתאימות מספרי הדף והשורה. המהדיר בודק את מיון השורות בזמן קליטת התוכנית להידור. סימון השורות מאפשר התיחסות המתכנת אליהן.

סדר חטיבות התוכנית ופרקיה חייב להשמר בהתאם לדוגמה זו. בפרק המתאר את הנתונים (WORKING STORAGE) מוגדרים ארבעה שדות מספריים עבור המונים, במבנה של שש ספרות שלמים ושתי ספרות עשרוניות, עם שמירת סימן (S). הערך ההתחלתי של השדות האלו הוא אפס, בהתאם לפרמטר של VALUE (שורות 1-08 עד 1-11).

החל משורה 1-12 מתואר מבנה הכרטיס המנוקב. יש פרוט רק ל-2 השדות המעניינים אותנו, אשר נמצאים בטורים הראשונים של הכרטיס. שדה IND הוא נומרי בן ספרה אחת ושדה ALEF הוא נומרי עם ארבע ספרות לשלימים ושתי ספרות עשרוניות. שאר טורי הכרטיס אינם חשובים ועל כן מוגדרים כ-FILLER. שדות אלו הם נתונים אלמנטריים של הנתון הקבוצתי KELET. שים לב לרמות השדה ולאופן כתיבתן. תאור השדות ניתן רק לנתונים אלמנטריים.

החל משורה 1-17 מפורטות הוראות התוכנית אשר ב-PROCEDURE DIVISION. בקטע START (שורה 1-17) מבוצעת קריאת הכרטיס. מיד לאחר הוראה זו ישנה הוראת הסיעוף המותנה (שורה 1-18): לבצע סיעוף אל הקטע שבו יבוצע החיבור - בהתאם לתוכן השדה IND. שים לב למבנה הוראת הסיעוף המותנה לפי ערך השדה. אם ב-IND ישנו ערך השונה מ-1, 2 או 3 לא יבוצע כל סיעוף והתוכנית תמשיך בהוראה העוקבת -

IBM COBOL Coding Form

SYSTEM: NEW PROGRAMS DEVELOPMENT — PUNCHING INSTRUCTIONS

PROGRAM: TARGIL — GRAPHIC

PROGRAMMER: MYSELF — DATE: 1975 — PUNCH

SEQUENCE | A B | COBOL STATEMENT | IDENTIFICATION

```
001010 IDENDIFICATION DIVISION.                                    TARGIL-A
001020 PROGRAM-ID. 'TARGIL'.                                       TARGIL-A
001030 AUTHOR. GOOD-PROGRAMMER NAMED AVIVI.                        TARGIL-A
001040 DATE-WRITTEN. YEAR 1975.                                    TARGIL-A
001050 ENVIRONMENT DIVISION.                                       TARGIL-A
  1060 DATA DIVISION.
  1070 WORKING-STORAGE SECTION.
  1080 77  MONE1   PIC  S9(6)V99  VALUE 0.
  1090 77  MONE2   PIC  S9(6)V99  VALUE 0.
  1100 77  MONE3   PIC  S9(6)V99  VALUE 0.
  1110 77  MONE4   PIC  S9(6)V99  VALUE 0.
  1120 01  KELET.
  1130        02  IND  PIC  9.
  1140        02  ALEF PIC  S9(4)V99.
  1150        02  FILLER PIC  X(73)
  1160 PROCEDURE DIVISION.
  1170 START.  ACCEPT  KELET.
  1180         GO TO  ADD1  ADD2  ADD3  DEPENDING ON IND.
  1190            ADD ALEF TO  MONE4.  GO TO  START.
  1200 ADD1.  ADD ALEF TO MONE1. GO TO START.
  1210 ADD2.  ADD ALEF TO MONE2. GO TO START.                      TARGIL-A
001220 ADD3.  ADD ALEF TO MONE3. GO TO START.                      TARGIL-A
```

כנ"ל

גנר'יסים

סיכום ל-MONE4 (שורה 1-19). לאחר מכן יבוצע סיעוף לא מותנה לראש הקטע START לקריאת כרטיס חדש.

בקטעים ADD1 , ADD2 , ADD3 מבוצע הסיכום של השדה ALEF למונה המתאים לערך שהיה ב-IND. לאחר מכן מבוצע סיעוף אל START (שורות 1-20, 1-21, 1-22).

שים לב שהתוכנית עדיין אינה שלימה בכך שאינה בודקת לכרטיס אחרון ולכן גם אינה מסתיימת. היא נמצאת במחזור אינסופי של קריאת כרטיסים ועיבוד. מקובל לכנות זאת מצב של לולאה (loop) אשר עלולה להגרם לפעמים ע"י שגיאת תכנות.

משפטי התנייה

מבנה משפט ההתנייה - IF

משפט התנייה מורה למחשב לבדוק ערכו של ביטוי או של משתנה ולבצע הוראות רצויות בהתאם לתוצאות הבדיקה. ההוראות שאותן מבצעים יכולות להיות פסוקי ביצוע של השפה ו/או הוראת סיעוף.

צורתו הכללית של משפט ההתנייה היא:

IF (תנאי) [THEN] (משפט קובול).

"משפט קובול" יתבצע רק כאשר התשובה ל"תנאי" תהיה חלובית, כלומר, כאשר התנאי מתקיים. אם המשפט כולל פסוקים שבסופם הוראת סיעוף - יבוצע סיעוף. ואם המשפט כולל הוראות ביצוע אחרות הן תבוצענה בזו אחר זו, ולאחריהן יבוצע המשפט העוקב.

כאשר התשובה ל"תנאי" היא שלילית, כלומר, כאשר התנאי איננו מתקיים יש דילוג על "משפט קובול" עד הנקודה שלאחרי הפסוק האחרון במשפט ההתנייה. ההוראה הבאה שתתבצע תהיה במשפט שלאחר נקודה זו.

מכאן, חובה שתופיע נקודה בסיומו של משפט התנייה (IF) מכיון שיש לה תפקיד לוגי משמעותי.

שים לב שהמלה THEN היא רשות, אינה חייבת להכתב במשפט ואפשר להשמיט אותה.

דוגמה:

```
MØVE ZERØ TØ MØNE.
IF (תנאי) ADD 5 TØ MØNE.
ADD 2 TØ MØNE.
```

הסבר: אם התנאי מתקיים, יבוצע הפסוק ADD 5 TO MONE, ולאחריו המשפט ADD 2 TO MONE. בסיום ביצוע קטע זה ערכו של המשתנה MONE יהיה 7.

אם התנאי לא מתקיים, לא יבוצע הפסוק ...ADD 5 ועל כן ערכו של MONE בסיום יהיה 2, מכיון שיתבצע דילוג במשפט התנאי עד לאחר הנקודה שבסופו.

פסוקי הקובול שבהוראת ה-IF יכולים להיות כל פסוק חוקי של השפה המורכב מהוראה אחת או מספר הוראות. לדוגמה:

```
a)  IF (תנאי) THEN MØVE A TØ B
                   DISPLAY A
                   ADD 3 TØ CC RØUNDED
                   GØ TØ KTØVET.
    DISPLAY 'TNAY SHAGUY' UPØN CØNSØLE.


b)  IF (תנאי) GØ TØ START.
    MØVE KTØVET TØ PRINT-AREA.
```

בדוגמה הראשונה בנוי "משפט קובול" ממספר פסוקים כשבסופם הוראת הסיעוף GO TO KTOVET. שים לב שלאחר הוראה זו מופיעה נקודה (.). הוראות אלו תתבצענה רק כאשר התשובה לתנאי היא חיובית. כאשר התשובה שלילית- תבוצע ההוראה DISPLAY 'TNAY ....

בדוגמה השניה משפט קובול הוא הוראת סיעוף בלבד. אם התנאי לא יתקיים תבוצע ההוראה ... MOVE.

התניות יחסיות מותרות

פסוק התנאי במשפט גורם להערכת מצב היחס בין שני משתנים או בין משתנה וליטרל מספרי.

א. יחסים בין שדות

את ביטוי היחס אפשר לציין בכמה צורות. בנוסף אפשר לצרף גם את הביטוי NOT להוראת השלילה.

	(שדה 2)	GREATER THAN	[NØT]	(שדה 1)
או:	(שדה 2)	>	[NØT]	(שדה 1)

(שדה 2) LESS THAN [NØT] (שדה 1)
או: (שדה 2) < [NØT] (שדה 1)

(שדה 2) EQUAL TØ [NØT] (שדה 1)
או: (שדה 2) = [NØT] (שדה 1)

<u>דוגמאות:</u>

IF SADE1 GREATER THAN SADE2 GØ TØ SØF.

אם SADE1 <u>גדול מ-</u>SADE2 יש להסתעף לקטע SOF.

IF MØNE < 0 THEN MØVE ZERØ TØ MØNE.

אם MONE <u>קטן מאפס</u> יש להעביר אפסים ל-MONE.

IF SADE3 NØT EQUAL TØ 4 GØ TØ LØØP.

אם SADE3 <u>איננו שוה</u> לערך (ליטרל) 4, יש להסתעף לקטע LOOP.

ב. <u>בדיקת סימן של שדות מספריים</u>

ערך השדה חיובי/לא חיובי	(שדה) [NØT] PØSITIVE
ערך השדה הינו אפס/איננו אפס	(שדה) [NØT] ZERØ
ערך השדה שלילי/לא שלילי	(שדה) [NØT] NEGATIVE

<u>דוגמאות:</u>

IF MØNE1 PØSITIVE THEN GØ TØ HEMSHECH.

אם ערכו של MONE1 <u>חיובי</u> יש להסתעף לקטע HEMSHECH.

IF CØUNTER-BØYS NØT ZERØ DISPLAY 'STØP' UPØN CØNSØLE.

אם ערכו של COUNTER-BOYS <u>איננו שוה לאפס</u>, יש להדפיס בקונסול את המלה STOP.

IF TØZAA NEGATIVE MØVE 0 TØ TØZAA.

אם השדה TOZAA הינו <u>שלילי</u> יש לאפס אותו.

ג. <u>בדיקת אופי הנתונים</u>

בדיקה אם השדה מספרי/לא מספרי	(שדה) [NØT] NUMERIC
בדיקה אם השדה אלפביתי/לא אלפביתי	(שדה) [NØT] ALPHABETIC

דוגמאות:

```
IF GIL NØT NUMERIC GØ TØ ERRØR.
```

אם השדה GIL אינו מספרי (כלומר מכיל גם אותיות או סימנים) יש להסתעף לקטע ERROR.

```
IF SHEM ALPHABETIC MØVE 1 TØ GØØD.
```

אם שדה SHEM הינו אלפביתי יש להעביר 1 למונה GOOD.

תנאי והיפוכו באותו משפט ... IF ... ELSE

IF (תנאי) [THEN] (משפט קובול1) ELSE (משפט קובול2).

משמעות המשפט:

אם התנאי מתקיים, יתבצע משפט קובול 1 ואם לא - יתבצע משפט קובול 2. ההוראה הבאה שתבוצע היא זו שלאחר הנקודה.

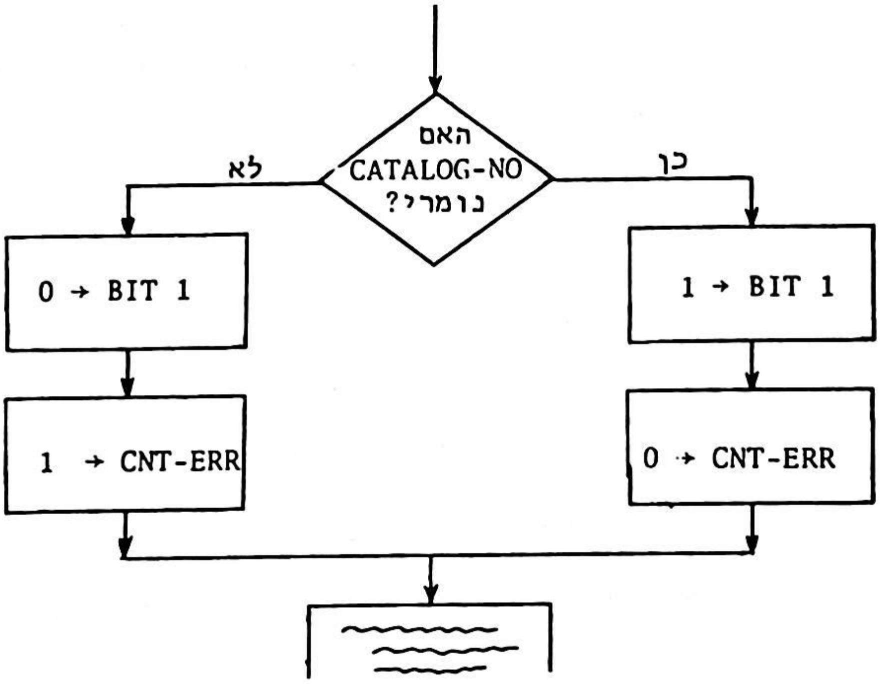
אם התנאי לא מתקיים, יתבצע משפט קובול 2 בלבד, וההוראה הבאה תהיה זו שאחרי הנקודה. כל משפט יכול להכיל פסוק אחד או יותר. המלה THEN אינה חובה ואפשר להשמיט אותה.

דוגמה:

```
IF CATALØG-NØ NUMERIC MØVE 1 TØ BIT1
   MØVE ZERØ TØ CNT-ERR
   ELSE
   MØVE ZERØ TØ BIT1
   MØVE 1 TØ CNT-ERR.
   ----------
```

שים לב: אופן הכתיבה המוצג כאן לפיו נכתבות הוראות בשורות נפרדות, אינו חובה אך נעשה לשם נוחיות תוך כדי ניצול אפשרות הכתיבה החפשית של ההוראות. כל ההוראות נכתבות באזור B, מטור 12 ואילך.

הדוגמה הנ"ל מתארת את המצב הבא:



התניות עם אופרטורים לוגיים

קיימת אפשרות לקשור במשפט התנייה מורכב מספר תנאים ע"י אופרטורים לוגיים של חיתוך (או - OR), איחוד (וגם.. - AND) ושלילה (NOT). במשפטי התנייה אלו האופרטורים הלוגיים הם בנוסף לאופרטורים היחסיים והבדיקות האחרות המותרות בקובול.

מבנה משפט ההתנייה:

IF (תנאי 1) OR/AND [NOT] (תנאי 2) (משפט קובול1)[ELSE (משפט קובול2)].

התשובה למשפט ההתנייה המורכב תהיה חיובית רק כאשר התנאים יתקיימו בהתאם ליחסים שבהם וביניהם, ורק אז יתבצע "משפט קובול 1".

כאשר משתמשים באופרטור החיתוך OR די שאחד משני התנאים הקשורים ביניהם באופרטור זה יתקיימו כדי שהתשובה תהיה חיובית ויבוצע "משפט קובול 1". כאשר משתמשים באופרטור האיחוד AND חייבים שני התנאים הקשורים ביניהם באופרטור זה להתקיים כדי שהתשובה תהיה חיובית ויבוצע "משפט קובול 1".

במשפט התנייה מורכב זה קיימות האפשרויות הבאות:

- אפשר לכתוב מספר תנאים בזה אחר זה כשביניהם אופרטורים לוגיים שונים. התשובה למשפט ההתנייה תהיה חיובית כאשר כל התנאים יתמלאו.
- אפשר להוסיף את האופרטור NOT, למשל A = B AND NOT C = D. כלומר: כאשר A = B מתקיים וכאשר C = D איננו מתקיים.
- אפשר להשתמש באופציה ...ELSE כמו במשפט התנייה רגיל.

דוגמאות:

א. IF A = B AND B = C MØVE ZERØ TØ MØNE.

העברת אפס ל-MONE תתבצע רק כאשר שני התנאים מתקיימים (בגלל האופרטור AND כלומר כש-A = B = C. שים לב שבתנאי אחד אפשר להשוות שני ערכים בלבד.

ב. IF A = B ØR B NØT = C MØVE ZERØ TØ MØNE.

במקרה זה די בכך שאחד התנאים יתקיים או ש-A = B ו/או ש-B ≠ C.

שים לב: התנאי הראשון והתנאי השני (וכן הלאה) חייבים להיות מותרים ע"י השפה כפי שתוארו בסעיף הקודם. לדוגמה:

א. IF A = B AND A = C GØ TØ SØF.

המשפט חוקי כי שני התנאים מותרים. כאשר שני התנאים יתמלאו יבוצע הסיעוף אל SOF.

ב. IF A = B AND = C GØ TØ SØF.

המשפט לא חוקי כי C = איננו תנאי מותר.

ג. IF A < 5 ØR PØSITIVE GØ TØ GØØD.

המשפט לא חוקי כי POSITIVE בפני עצמו איננו תנאי מותר. המשפט הזה צריך להכתב כך:

IF A < 5 ØR A PØSITIVE GØ TØ GØØD.

כלומר: כאשר A קטן מ-5 או A הוא חיובי יבוצע סיעוף אל GOOD.

סדר הפעולות במשפט התנייה מורכב:

במקרה של משפט מורכב המכיל את האופרטור AND וגם את האופרטור OR יופעל תמיד האופרטור AND לפני האופרטור OR. אם רוצים לקבוע את התהליך הלוגי בשונה מכלל

זה - אפשר להשתמש בסוגריים. כלל זה דומה לכלל האלגברי לשימוש בסוגריים.

לדוגמה:

IF A > B ØR MØNE = 0 AND SADE NEGATIVE ...

במשפט זה יבוצע תחילה ה-AND ולאחר מכן ה-OR. אפשר להציג זאת בעזרת סוגריים בצורה הבאה:

IF (A > B) ØR ((MØNE = 0) AND (SADE NEGATIVE)) GØ TØ YØFI.

כלומר: אם A גדול מ-B או (OR)
אם MONE = 0 וגם (AND) הערך של SADE הוא שלילי,
רק אז אפשר להסתעף לקטע YOFI.

תרגיל מסכם לפעולות חשבוניות ומשפטי התנייה

תאור הבעיה

נתון קובץ כרטיסים המתאר תנועות בחשבון לקוחות של מוסד כלשהוא. תנועות תשלום או החזר המתיחסות לתשלום מאופיינות על ידי קוד פעולה מתאים. יש לבצע בדיקות לוגיות של נתוני הקלט, לחשב את היתרה החדשה לכל חשבון ולהפיק דוח מפורט לכל חשבון ושורת סיכום לכל הדוח.

להלן פרוט הבעיה:

א. מבנה כרטיסי הקלט:

טורים 1- 6 מספר לקוח
טור 7 קוד כרטיס (1 או 2)
טורים 8-15 יתרה קודמת (שתי ספרות עשרוניות)
טור 16 קוד פעולה (1 - החזר)
(2 - תשלום)
טורים 17-22 סכום תשלום/החזר (שתי ספרות עשרוניות)

הכרטיס האחרון בקובץ יצויין ע"י * (כוכבית) בטור 1. שארית הכרטיס לא תכיל נתונים.

ב. כל כרטיס (רשומה) יבדק בדיקות לוגיות. יש לדלג על כרטיסים שגויים ורק לספור אותם לשם דוח בסיום התוכנית.
הבדיקות הלוגיות של הקלט הן:

- השדות מספר לקוח, תשלום/החזר - צריכים להיות מספריים,
- שדה קוד כרטיס צריך להכיל את הספרות 1 או 2,
- שדה יתרה קודמת צריך להיות חיובי.

ג. לכל כרטיס שעבר את הבדיקות בהצלחה תחושב יתרה חדשה לפי הנוסחה הבאה:

יתרה חדשה = יתרה קודמת + (פלוס) תשלום.

או: יתרה חדשה = יתרה קודמת - (מינוס) החזר.

משמעות הסכום בשדה סכום תשלום/החזר נקבעת לפי הקוד שבשדה קוד - פעולה.
אם שדה היתרה החדשה הוא שלילי יש להדפיס הודעת שגיאה למפעיל, לדלג עליו, ולא להעבירו לסיכום היתרות ולהדפסת הפלט.

ד. הפלט של תוכנית זו יכלול שתי קבוצות דווח (במדפסת).

1) דוח יתרות חדשות אשר בו שורה לכל לקוח. כל שורה תכיל מספר חשבון והיתרה החדשה (שתחושב).

2) שורת סיכום בסוף הדוח אשר תכיל את הפרטים הבאים: סה"כ כרטיסים בקובץ הקלט, סה"כ כרטיסים שגויים, סה"כ (סכום) יתרות חדשות.

פתרון הבעיה

א. תאור הקלט והפלט והגדרת שדות לתכנות

1) מבנה הכרטיס

מספר לקוח	קוד כרטיס	יתרה קודמת	קוד פעולה	סכום תשלום/החזר	
1_ _ _6	7	8... 15	16	17_ _ _ 22	_ _ _

מספר לקוח - LAKUAH

קוד כרטיס - KOD

יתרה קודמת - YETRA (ללא נקודה עשרונית ממשית)

קוד פעולה - KOD-PEULA

סכום תשלום/החזר - SHUM (ללא נקודה עשרונית ממשית)

2) שורת הדפסה של דוח היתרות (שים לב למיקום הנקודה העשרונית):

יתרה חדשה:



3) <u>שורת הדפסה של סיכום היתרות</u>

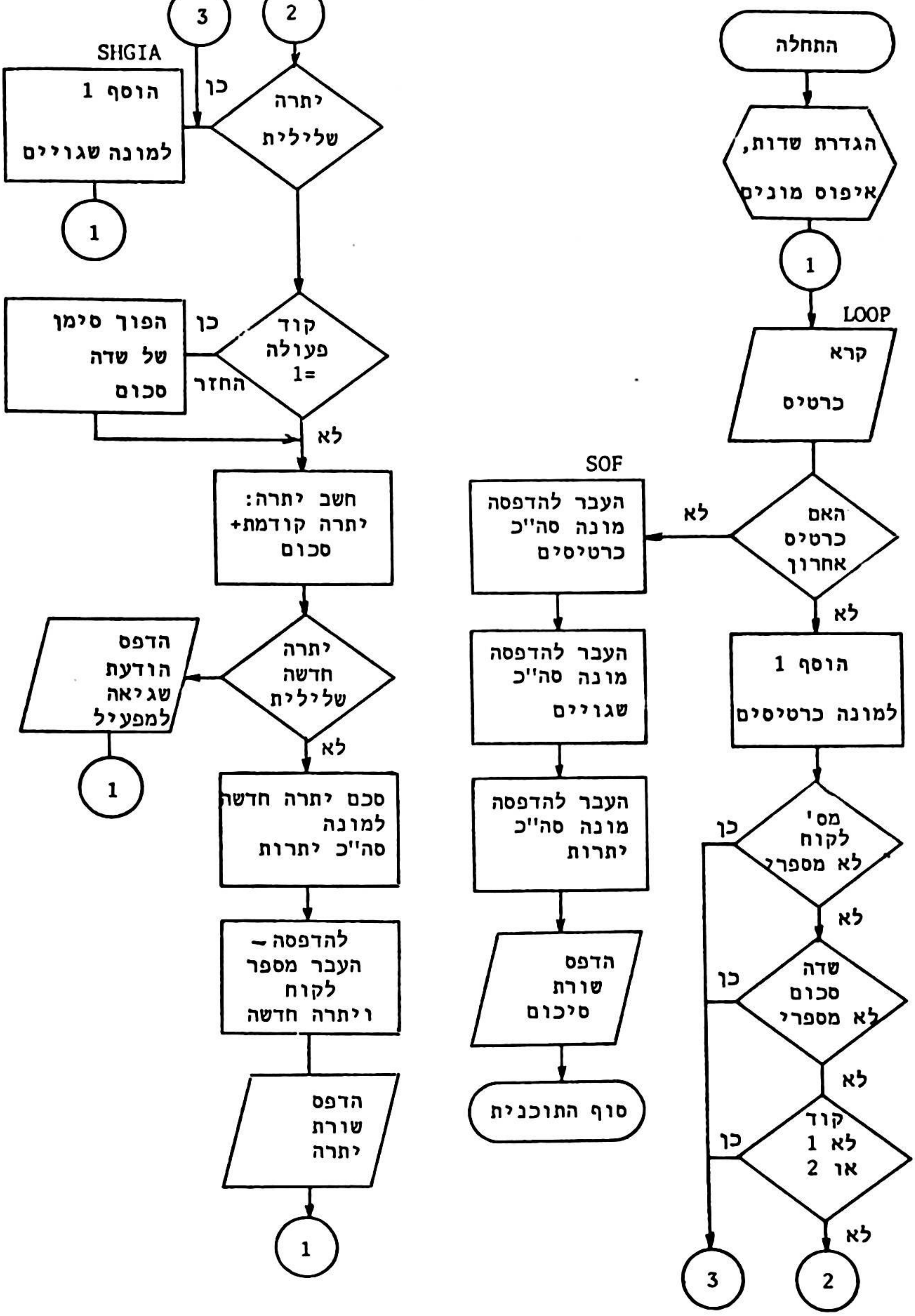


4) <u>שורת הדפסה של הודעת שגיאה</u>



<u>הערה</u>: "מעל" שורות ההדפסה נרשמו הקבועים שיודפסו על ידי התוכנית.
"מתחת" לשורה רשומים שמות השדות שערכיהם יועברו לצורך הדפסה.

ב. תרשים מלבני



ג. הסבר לתוכנית קובול

1) התוכנית כתובה בטפסי הקידוד של קובול. הקפד גם אתה לרשום כל תו במשבצת נפרדת, להקפיד על מרווחים, לכתוב באותיות גדולות, לרשום את המשפטים בצורה נכונה באזורים A ו-B (טורים 8, 12 בהתאמה) וגם - לרשום נקודה בסוף כל משפט. כל שורת קידוד מסומנת במספר סידורי.

2) סדר ה-DIVISIONS בתוכנית הינו קבוע:

- IDENTIFICATION
- ENVIRONMENT
- DATA
- PROCEDURE

שמות כולם חייבים להופיע גם כאשר אין פירוט ב-DIVISION מסוים, כמו למשל בשורה 1-03 (אשר טרם נלמד).

3) הגדרת שדות ורשומות ניתנת ב-WORKING-STORAGE SECTION (שורה 1-05). את השדות האלמנטריים ברמה 77 (שורות 1-06 עד 1-09) מגדירים לראשונה, לפני הגדרת רשומות או נתונים קבוצתיים אחרים.

4) שדות המיועדים לשמש כמונים (שורות 1-06 עד 1-08), וחייבים להתחיל עם ערך אפס, יוגדרו כ-VALUE ZERO. אם אינם מוגדרים בצורה זו, ערכם ההתחלתי אינו ידוע בתחילת התוכנית. אפשר לאפס מונים אלו בתחילת התוכנית (PROCEDURE DIVISION) ע"י ההוראה ... MOVE ZERO TO.
שדה YETRA-HADASHA מחושב במהלך התוכנית ועל כן אין צורך לאפס אותו תחילה.
שדות אלו מוגדרים כנתונים אלמנטריים ברמה 77.

5) הנתונים הקבוצתיים שהוגדרו ברמה 01 הם רשומת הקלט (מבנה הכרטיס), KARTIS, רשומת ההדפסה של הודעת שגיאה על יתרה שלילית - HODAHA, רשומת ההדפסה של היתרה החדשה לכל לקוח - SHURA, ושורת הסיכום של הדוח - SHURAT-SIKUM.
בכל נתון קבוצתי קיימים נתונים ברמת שדה גבוהה יותר (רמה אחת בלבד). בשדות אלו ניתן תאור השדה. בשורת הסיכום של הדוח יש הגדרת ערך התחלתי של שדות כמרווחים (SPACE) וקבועים אלפביתיים בהתאם למבנה שורת ההדפסה (שורות 1-20, 2-03, 2-06, 2-08, 2-10).

6) בשורה 1-12 נמצאת הגדרה של שדה קיים ע"י משפט REDEFINES. ההגדרה הזו יכולה להתיחס רק לחלק מהשדה שאותו מגדירים מחדש: השדה LAKUAH הוא בן 6 תוים וההגדרה החדשה (או הנוספת) מתיחסת רק לספרה אחת, השמאלית ביותר, המיועדת לבדיקת "כרטיס אחרון".

7) בשורות ההדפסה ישנם שדות עם תוי עריכה (1-21, 2-04, 2-11). השדה שאותו עורכים הוא שדה עם ערך כספי המכיל בתוכו נקודה עשרונית ממשית. בהדפסה רוצים לבטל את האפסים המובילים ולכן ישנו תו העריכה Z משמאל. בשורה 1-21 ישנו תו עריכה מינוס (-) המאפשר הדפסת סימן "שלילי" ליד הערך המספרי.

8) תוכנית הביצוע מתחילה ב- PROCEDURE DIVISION, בהוראת קריאה של כרטיס אל הרשומה KARTIS ובדיקה אם זהו הכרטיס האחרון בקובץ הנושא את הסימן * (שורות 2-13, 2-14). אם הוא האחרון - עוברים לקטע הסיום של התוכ-נית - SOF (שורה 3-10). אם הכרטיס אינו אחרון מוסיפים 1 למונה הכר-טיסים ומתחילים בבדיקות הלוגיות.

9) הבדיקות הלוגיות מתוארות בשורות 2-15 עד 2-18. כאשר מתגלית שגיאה עוברים לקטע התוכנית SHGIA (3-08). בקטע זה סופרים את מספר השגיאות וחוזרים לקריאת כרטיס חדש (2-13). שים לב לאופן כתיבת התנאים המור-כבים: בדיקת שני תנאים בהוראה אחת.

10) כל הכרטיסים שאינם שגויים עוברים להמשך העיבוד, החל משורה 2-19. בודקים את סוג הפעולה כדי לדעת איך לחשב את היתרה החדשה. מכיון שבהוראה זו אין סיעוף, הרי בכל מקרה תבוצע הוראת COMPUTE (2-20) כלומר אם התשובה ל-IF היא חיובית וגם אם היא שלילית.
הכפלת השדה SHUM במספר (-1) באה כדי להפוך את הסימן שלו ולאפשר חי-שוב היתרה החדשה כמתואר בשורה 2-20. כלומר, אם השדה SHUM מייצג החזר הערך שלו יהפך לשלילי ויסוכם (כלומר בסימן -) אל שדה יתרה קודמת.
בשלב זה עורכים בדיקה אם היתרה החדשה שלילית (2-21). אם היא שלילית מעבירים את מספר הלקוח והיתרה לשורת ההדפסה של השגיאה (2-22, 2-23). מדפיסים הודעה למפעיל ועוברים לקריאת כרטיס חדש (לשורה 2-13). אם היתרה אינה שלילית (כלומר, חיובית) מדלגים על כל ההוראות שבמשפט IF (עד הנקודה) וממשיכים בהוראה העוקבת בשורה 3-03.

11) את היתרה החדשה מסכמים אל שדה סה"כ יתרות (3-03) לאחר החישובים מעבירים את שדות מספר לקוח ויתרה חדשה (שחושבה) מעבירים לשדות

COBOL טופס קרוד

SYSTEM: COBOL LESSON
PROGRAM: SAMPLE
PROGRAMMER: DANY　DATE: 1975

```
IDENTIFICATIØN DIVISIØN.
PRØGRAM-ID. 'TARGIL'.
ENVIRØNMENT DIVISIØN.
DATA DIVISIØN.
WØRKING-STØRAGE SECTIØN.
77  MØNE-KLALI       PIC  9(4)      VALUE ZERØ.
77  MØNE-SHAGUI      PIC  9(4)      VALUE ZERØ.
77  SAH-YETRØT       PIC  S9(7)V99  VALUE ZERØ.
77  YETRA-HADASHA    PIC  S9(6)V99  VALUE ZERØ.
01  KARTIS.
    03  LAKUA   PIC 9(6).
    03  FIRST-CØLUMN REDEFINES LAKUAH PIC X.
    03  KØD     PIC X.
    03  YETRA   PIC S9(6)V99.
    03  KØD-PEULA PIC 9.
    03  SHUM    PIC S9(4)V99.
    03  FILLER  PIC X(52).
01  HØDAHA.
    02  D-LAKUAH   PIC 9(6).
    02  FILLER     PIC X(10) VALUE '  SHLILY :'.
    02  D-YETRA    PIC ZZZZZ9.99-.
01  SHURA.
    05  LAKUAH-Ø    PIC 9(6).
    05  FILLER      PIC XX VALUE SPACE.
    05  NEW-YETRA-Ø PIC ZZZZZ9.99.
01  SHURAT-SIKUM.
    03  FILLER   PIC X(10) VALUE 'SAH HAKOL-'.
    03  SAH-Ø    PIC 9(4).
    03  FILLER   PIC X(10) VALUE '   SHGUIM-'.
    03  SHG-Ø    PIC 9(4).
    03  FILLER   PIC X(10) VALUE '   YETRØT-'.
    03  YTRØT-Ø  PIC ZZZZZZZ9.99.
PRØCEDURE DIVISIØN.
LØØP.   ACCEPT KARTIS. IF FIRST-CØLUMN = '*' GØ TØ SØF.
            ADD 1 TØ MØNE-KLALI.
        IF LAKUAH NØT NUMERIC ØR SHUM NØT NUMERIC
           GØ TØ SHGIA.
        IF KØD < 1  ØR  KØD > 2  GØ TØ SHGIA.
        IF YETRA NEGATIVE  GØ TØ SHGIA.
        IF KØD-PEULA = 1  MULTIPLY -1 BY SHUM.
           CØMPUTE  YETRA-HADASHA = YETRA + SHUM.
        IF YETRA-HADASHA NEGATIVE
           MØVE LAKUAH TØ D-LAKUAH
           MØVE YETRA-HADASHA TØ D-YETRA
        DISPLAY  HØDAHA  UPØN CØNSØLE
           GØ TØ LØØP.
           ADD YETRA-HADASHA TØ SAH-YETRØT.
           MØVE LAKUAH TØ LAKUAH-Ø.
           MØVE YETRA-HADASHA TØ NEW-YETRA-Ø.
        DISPLAY SHURA.
           GØ TØ LØØP.
SHGIA.  ADD 1 TØ MØNE-SHAGUI.
           GØ TØ LØØP.
SØF.       MØVE MØNE-KLALI TØ  SAH-Ø.
           MØVE MØNE-SHAGUI TØ SHG-Ø.
           MØVE SAH-YETRØT TØ  YTRØT-Ø.
        DISPLAY SHURAT-SIKUM.
           STØP RUN.
```

בשורת ההדפסה (3-04, 3-05), מדפיסים את השורה (3-06) ומסתעפים להוראת הקריאה (3-07) לקריאת כרטיס חדש (2-13).

(12) קטע סיום התוכנית (שורות 3-10 עד 3-14) מתבצע כאשר מתגלה הכרטיס האחרון. כאן עורכים את המונים בשדות של שורת הסיכום של הדוח, מדפי-סים את השורה (3-06) ולבסוף - מסיימים את התוכנית ע"י ההוראה STOP RUN.

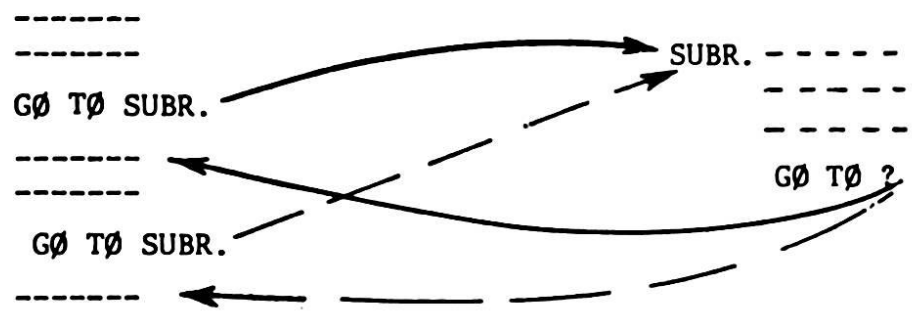
(13) הערה: בפתרון זה השתמשנו בהוראות ACCEPT ו-DISPLAY. הוראות אלו מיו-עדות בדרך כלל לקריאת כרטיסים בודדים ולהדפסת שורות בודדות. ראה גם פרק "קלט פלט בקבצים".

שגרות - Subroutines

השגרה בתוכנית

שגרה או סברוטינה הינה קטע בתוכנית אשר מבצעים אותו פעמים רבות, ומגיעים אליו בדרך כלל ממקומות שונים בתוכנית. השונה בין שגרה לבין קטע תוכ-נית אחר הוא בכך שלאחר ביצוע השגרה מתכוונים לחזור לאותו מקום בתוכנית ממנו בוצעה ההסתעפות אליה.

את מסלול ביצוע ההוראות אפשר לתאר בצורה הבאה:



כלומר לאחר ביצוע השגרה חוזרים להוראה העוקבת להוראת הסיעוף אליה, או אל כל הוראה אחרת בתוכנית.

הבעיה המתעוררת בניהול השגרה הינה קביעת כתובת הסיעוף (שם קטע) בהו-ראה האחרונה שלה, המכוונת להחזיר את הפיקוח אל ההוראה הרצויה למשתמש. כלומר - דרוש מנגנון לשנוי דינמי של הכתובת בהוראת הסיעוף.

ההוראה ALTER מאפשרת לשנות את כתובת הסיעוף לכל כתובת רצויה בתוכנית.

מכיון שברוב המקרים מתכוונים לחזור להוראה העוקבת בלבד, ישנה הוראת קובול מיוחדת לצורך זה - ההוראה PERFORM. הוראה זו חוסכת מהמתכנת את טרחת כתיבת הוראת הסיעוף בסוף השגרה והשנוי של הוראה זו בכל פעם שצריך לבצע סיעוף אל השגרה.

ההוראה - ALTER

ההוראה ALTER מאפשרת לשנות את כתובת הסיעוף ביציאה מהשגרה אל כל כתובת רצויה. לאחר ביצוע הוראת השנוי של כתובת היציאה הזו, מבוצעת הוראת סיעוף אל השגרה:

ALTER (שם קטע של הוראת הסיעוף ביציאה מהשגרה) TØ PRØCEED TØ (שם קטע רצוי).

GØ TØ (שם קטע שגרה).

לדוגמה:

```
     A   B
1        ALTER SØF-SUBR TØ PRØCEED TØ BACK1.
2              GØ TØ SUBRØUTINE.
3    BACK1. MØVE ----
4           ---------
5        ALTER SØF-SUBR TØ PRØCEED TØ BACK2.
6              GØ TØ SUBRØUTINE.
7    BACK2. ADD ----
8           --------
9    SUBRØUTINE. MØVE ----
10          --------
11   SØF-SUBR. GØ TØ SØF-SUBR.
```

הסבר: השגרה מתחילה בשורה 9. כתובת היציאה שלה (בשורה 11) מכוונת אל עצמה כמצב התחלתי, אשר במהלך התוכנית תתבצע הוראת ALTER (בשורה 1) אשר תשנה את כתובת היציאה של השגרה ל-BACK1 , ולאחר מכן יתבצע סיעוף לסברוטינה (בשורה 9). בסיום השגרה יבוצע סיעוף ל-BACK1 (שבשורה 3). במהלך התוכנית תשנה הוראת ALTER (בשורה 5) את כתובת היציאה של השגרה ל-BACK2, ואח"כ יתבצע סיעוף. לאחר סיום השגרה תחזור התוכנית לקטע BACK2.

הערה: ההוראות ALTER ו- GØ TØ אינן חייבות להיות צמודות.

הוראה PERFORM הבסיסית לקטע אחד

הוראה PERFORM מאפשרת לבצע סיעוף לשגרה ולחזור להמשך ביצוע התוכנית ללא שימוש בהוראות ALTER ו-GØ TØ.

מבנה ההוראה:

PERFORM (שם קטע).

ההוראה מבצעת סיעוף אל הקטע המצויין בהוראה. לאחר השלמת ביצוע כל ההוראות הכלולות בקטע זה תבוצע הוראת סיעוף של התוכנית לבצע ההוראה העוקבת את הוראת ה-PERFORM, ולאחריה תמשיך בסדר ההוראות. הוראת PERFORM מוסיפה, ביוזמתה, את הוראת הסיעוף הדרושה בסוף הקטע המוגדר בהוראה.

הדרישה היחידה לקיום פעילות הוראה זו היא שהקטע שאליו מבוצע סיעוף לא יכלול הוראות סיעוף אל מחוץ לקטע, אך אפשרי סיעוף אל ראש הקטע עצמו. אם קיימת הוראת סיעוף אל מחוץ לקטע, ואם היא מתבצעת (למשל כתשובה חיובית להוראת סיעוף מותנה) - יפסק מחזור ה-PERFORM והתוכנית תמשיך לפי כתובת הסיעוף בקטע האחר, ולא תחזור להוראה שלאחר ה-PERFORM. הסיבה לכך-ההוראה מכירה קטע אחד בלבד, ומכיון שהסיעוף התבצע אל קטע אחר הוא איננו נמצא בשליטתה.

דוגמה א':

```
          PERFØRM KETA-1.
          ADD 1 TØ MØNE.
          -------
KETA-1.   ADD ----
          MØVE ----
KETA-2.   CØMPUTE ----
          --------
```

הוראת PERFORM תגרום לביצוע סיעוף אל KETA-1. בקטע זה תתבצענה שתי ההוראות שמרכיבות אותו. ההוראה הבאה שתתבצע היא ADD 1 TO MONE הרשומה מיד לאחר ההוראה PERFORM.

שים לב שמשפט PERFORM כולל הוראה זו בלבד.

דוגמה ב':

```
1        PERFORM CALC.
2        MØVE 1 TØ FIELD.
         -----
3   CALC. MØVE ----
4         ADD  ----
5         IF SADE = 0 GØ TØ CHECK.
6         ADD  ----
7         CØMPUTE ----
8   ØUTPUT. ----
            ----
9   CHECK.  ----
            ----
```

הסבר: ההוראה PERFORM (שורה 1) תגרום לסיעוף אל הקטע CALC (שורה 3) ולביצוע הפקודות הכלולות בו: לאחר ביצוע הפקודות MOVE ו-ADD (שורות 4,3) יבוצע משפט ההתנייה IF (שורה 5). אם התנאי לא מתקיים (כלומר SADE ≠ 0) יתבצעו ההוראות שלאחר הוראת IF (שורות 6, 7) והתוכנית תחזור להוראה שלאחר ה-PERFORM בשורה 2, ותמשיך.

אולם, אם התנאי מתקיים, התוכנית תבצע סיעוף לקטע CHECK (שורה 9) ותצא ממסגרת הקטע שהוגדר להוראת ה-PERFORM. התוכנית תבצע את ההוראות בקטע CHECK ותמשיך הלאה בהתאם להוראות ולא תחזור להוראה שלאחר ה-PERFORM, לשורה 2.

הערה: הקטע אליו מסתעפים ע"י ההוראה PERFORM הינו קטע רגיל לכל דבר ולכן אפשר גם ל"עבור" דרכו במהלך ביצוע סקונציאלי של הוראות.

לדוגמה:

```
1        MØVE ZERØ TØ MØNE.
2        PERFØRM KETA-1.
3        ADD 1 TØ MØNE.
4   KETA-1.
5        ADD 2 TØ MØNE.
6   SØF. STØP RUN.
```

הסבר: מאפסים את MONE (שורה 1). הוראת PERFORM מבצעת את ההוראות אשר כלולות ב-KETA-1 (שורות 5-4). ההוראה בשורה 5 מכוונת להוסיף 2 ל-MONE. עם סיומה

חוזרים להוראה העוקבת את הוראת PERFORM, כלומר לשורה 3. הוראה זו מכוונת להוסיף 1 ל-MONE (אשר כעת ערכו הוא 3). ההוראה העוקבת שייכת ל-KETA-1 והיא מוסיפה 2 ל-MONE (שורה 5). עם סיום קטע התוכנית בשורה 6 ערכו של MONE הוא 5.

שים לב: בדוגמה זו ההוראה STOP RUN חייבת לקבל "שם קטע" (למשל - SOF) כדי להפריד אותה מהקטע הקודם (KETA-1) אשר מתבצע כשגרה. אם לא היה להוראה זו "שם-קטע", הרי שבזמן ה-PERFORM היינו מגיעים להוראה זו והתוכנית היתה מפסיקה, ואז תוכנו של MONE היה 2.

ההוראה PERFORM עם תחום קטעים

לעתים מורכבת השגרה מסדרה של קטעים רצופים המהוים לצורך מסויים יחידה לוגית אחת. אפשר לבצע קטעים אלו כשגרה אחת על ההוראה המתאימה:

PERFORM (שם קטע1) THRU (שם קטע2).

ההוראה גורמת לסיעוף ל"שם קטע 1". החזרה לפקודה העוקבת תהיה רק לאחר גמר "קטע 2". מספר הקטעים יכול להיות שניים ויותר, אך הם חייבים להיות רצופים, וללא סיעופים אל מחוץ להם, מאותה סיבה שהוזכרה בהוראת PERFORM הבסיסית.

דוגמה:

```
    PERFØRM AA THRU CC.
    --------

AA. CØMPUTE SADE = SHUM - YETRA.
    IF SADE NEGATIVE GØ TØ BB.
    MØVE 1 TØ SIMAN. GØ TØ CC.

BB. MØVE ZERØ TØ SADE.
    MØVE 2 TØ SIMAN.

CC. DISPLAY 'STØP3' UPØN CØNSØLE.
DD. --------
```

הסבר: הוראת PERFORM מכוונת לביצוע הקטעים AA, BB, CC. הסיעוף מתבצע אל הפקודה הראשונה בקטע הראשון - AA. עם סיום ההוראות בקטע CC תבוצע חזרה אל ההוראה שלאחר ה-PERFORM. שים לב שישנה אפשרות לבצע סיעופים בין קטע אחד

לשני: בקטע AA ישנה הוראת הסיעוף המותנה המאפשרת סיעוף לקטע BB, וגם הוראת סיעוף ללא התנייה - לקטע CC. אולם בכל מקרה - היציאה האחרונה צריכה להיות מסוף קטע CC.

הוראת הסרק - EXIT

בכתיבת שגרה הכוללת מספר קטעים יש צורך לעתים בקטע אחרון, פיקטיבי, שאינו כולל הוראות. קטע זה מיועד רק לנקז אליו הוראות סיעוף שונות המכוונות אל כתובת היציאה של השגרה.

בקטע זה צריכה להיות הוראה כלשהיא שאינה מבצעת דבר. הוראת סרק זו הינה ההוראה EXIT.

לדוגמה:

```
        PERFØRM AA THRU BB.
        ---------

    AA. CØMPUTE ----
        ADD ----
        IF A > B GØ TØ BB.
        ----------

    BB. EXIT.
    CC. ----------
```

במקרה זה, כש-A > B לא דרושה כל פעולה נוספת בשגרה אלא להסתעף לכתובת היציאה שלה, וממנה לחזור להוראה הבאה לאחר ה-PERFORM. היציאה מהשגרה מבוצעת בקטע BB(...THRU BB) אך מכיון שאין צורך לבצע כל פעולה שהיא בקטע זה, משתמ-שים בהוראת ה-EXIT.

ביצוע חוזר לפי מונה

PERFORM (שם קטע1) [THRU (שם קטע2)] n TIMES.

הוראה זו מאפשרת לחזור על ביצוע שגרה מספר פעמים לפי הערך - n. רק לאחר השלמת ביצוע השגרה מספר פעמים כנדרש, תחזור התוכנית להוראה שלאחר ה-PERFORM.

השגרה יכולה לכלול קטע אחד בלבד או מספר קטעים אם משתמשים בפרמטר -

THRU (שם קטע 2). פרמטר זה הינו רשות.

<u>דוגמה:</u>

```
1           MØVE ZERØ TØ MØNE.
2           PERFØRM SIKUM 5 TIMES.
3           ADD 1 TØ MØNE.
4           STØP RUN.
5   SIKUM.  ADD 2 TØ MØNE.
```

<u>הסבר</u>: מאפסים את MONE (שורה 1). הוראת PERFORM (שורה 2) תגרום ביצוע הקטע SIKUM חמש פעמים. ההוראה בקטע SIKUM (שורה 5) מוסיפה 2 ל-MONE ולכן ביצועה 5 פעמים יגרום שבמונה יהיה הערך 10. ההוראה הבאה לביצוע היא בשורה 3 (ההוראה שלאחר ה-PERFORM) והיא תוסיף 1 ל-MONE. עם סיום התוכנית בהוראה בשורה 4 הערך של MONE יהיה 11.

מספר הפעמים לביצוע השגרה יכול להיות ערך קבוע כבדוגמה זו, או <u>דינמי</u> לפי ערכו של שדה מוגדר. כלומר, במקום ערך מספרי קבוע יש לרשום שם-שדה שאופיו מספרי. מספר הפעמים שהשגרה תתבצע מותנה <u>בתוכן</u> השדה <u>ברגע הביצוע</u>. את ערכו (תוכנו) של השדה אפשר לשנות באופן דינמי במהלך ביצוע התוכנית.

<u>דוגמה:</u>

```
77 CØUNT PICTURE 99.
--------
PERFØRM AA THRU BB CØUNT TIMES.
```

<u>ביצוע בהתנייה</u>

PERFØRM (שם קטע1) [THRU (שם קטע2)] UNTIL (תנאי).

הוראה זו משמשת לביצוע חוזר של שגרה <u>עד</u> לקיומו של התנאי המוגדר בהוראה. התנאי הוא כל תנאי המותר במשפט ה-IF.

בדיקת התנאי נעשית <u>לפני</u> כל הסתעפות לשגרה.

<u>דוגמאות:</u>

```
PERFØRM KETA UNTIL MØNE > 3.
PERFØRM KETA UNTIL SADE NEGATIVE.
PERFØRM KETA UNTIL A = B.
```

<u>הסבר</u>: כאשר התוכנית מגיעה להוראתPERFORM היא בודקת אם התנאי מתקיים. אם הוא לא מתקיים, יבוצע סיעוף ל-KETA. עם סיום ביצוע ההוראות שב-KETA תחזור התוכנית להוראתPERFORM ותבדוק אם התנאי מתקיים - אם <u>איננו</u> מתקיים יבוצע שוב סיעוף וחוזר חלילה.

אם התנאי מתקיים בהגיע התוכנית להוראת ה-PERFORM, (גם בפעם הראשונה), לא יתבצע שום סיעוף והתוכנית תמשיך בהוראה הבאה לאחר הוראת ה-PERFORM.

<u>ביצוע בהתנייה של אינדקס</u>

<u>PERFØRM</u> (שם קטע 1) [THRU (שם קטע 2)]
VARYING (שם שדה אינדקס)

FRØM $\left\{\frac{n_1}{\text{שם שדה לערך}}\right\}$

BY $\left\{\frac{n_2}{\text{שם שדה לקידום}}\right\}$

UNTIL (תנאי).

בהוראתPERFORM זו ההתנייה לביצוע מורכבת משני אלמנטים: אינדקס משתנה ותנאי בדיקה שלו.

ההוראה קובעת תוכן ראשוני לשדה האינדקס המוגדר ב-VARYING ובודקת את קיום התנאי. אם התנאי <u>מתקיים</u> - עוברת התוכנית להוראה שלאחר ה-PERFORM. אם התנאי <u>לא מתקיים</u> מבוצע סיעוף לשגרה הכוללת קטע אחד או יותר כמפורט בהוראת PERFORM. לאחר מכן, מקודם האינדקס והתוכנית חוזרת לבדיקת קיום התנאי, וחוזר חלילה עד לקיום התנאי. את שדה האינדקס יש להגדיר כנתון אלמנטרי באופי מספרי.

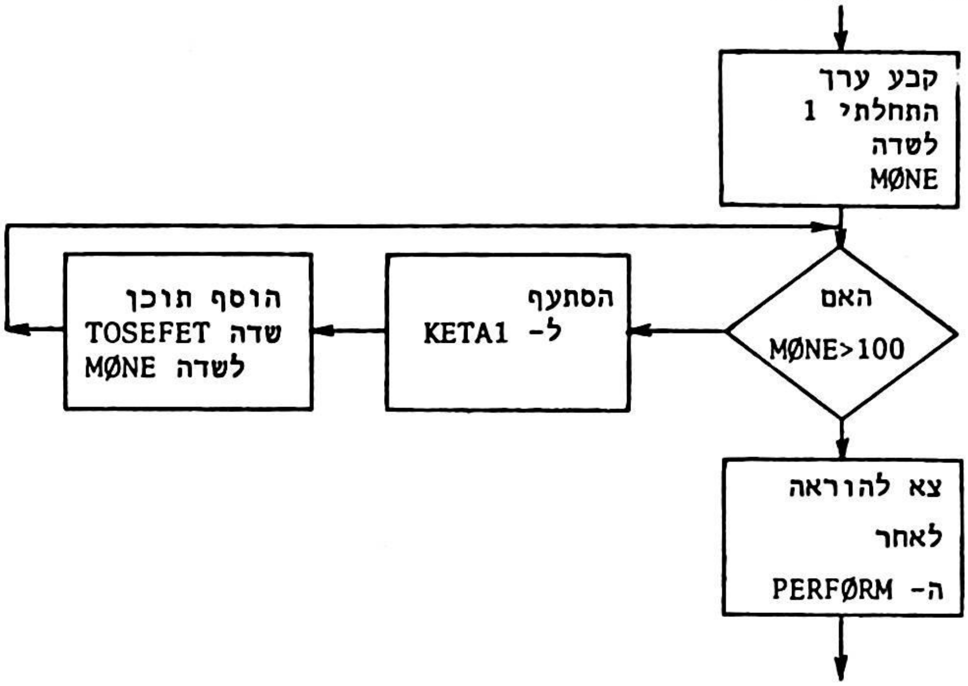
הערך ההתחלתי של האינדקס נקבע ב-FROM: אם צויין מספר - זהו הערך ההתחלתי של האינדקס ואם צויין שם שדה, <u>תוכנו</u> של השדה יהיה הערך ההתחלתי.

הקידום של האינדקס נקבע לפי הערך הנמסר ב-BY: אם צויין מספר - זהו הערך שאותו יש להוסיף בכל פעולת קידום של האינדקס, ואם צויין שם שדה - יש לקחת את תוכנו.

<u>דוגמה א'</u>:

```
77 MØNE PIC 99.
77 TØSEFET PIC 9.
 ----------
 PERFØRM KETA1 VARYING MØNE FRØM 1 BY TØSEFET
        UNTIL MONE > 100.
```

הסבר: מסלול ההוראה מתואר בצורה גרפית:



שים לב ששדה TOSEFET מוגדר כשדה שאופיו מספרי.

דוגמה ב':

```
77 INDEX PIC 99.
77 START-NØ PIC 99.
   -------
   PERFØRM KETA VARYING INDEX
           FRØM START-NØ
           BY 3
           UNTIL INDEX > 27.
```

הסבר: לפני ההסתעפות הראשונה ייקבע ערך התחלתי של השדה INDEX עפ"י תוכנו של השדה START-NO, אשר אופיו מספרי. לאחר השלמת ביצוע הקטע בפעם הראשונה ובפעמים הבאות, יתווסף לשדה INDEX המספר 3. ההסתעפות ל-KETA תופסק כשהערך של INDEX יהיה גדול מ-27.

הערות: 1. אופציה זו משמשת במיוחד לעבודה בטבלאות כפי שיוסבר בפרק הבא.
2. קיימת אופציה מורחבת של ההוראה לטיפול בשניים או שלשה אינדקסים. אופציה זו יש ללמוד בספרות המתאימה.

ט ב ל א ו ת

תאור ומבנה

טבלה היא אוסף של מספרים, שמות או ערכים הקשורים לנושא אחד. מרכיבי הטבלה נקראים איברים והם יכולים להכיל נתון אחד, או יותר, לפי המקרה. איברי הטבלה מסודרים בסדר כלשהוא: בלתי ממויינים, ממויינים ברשימה (טבלה חד-ממדית) או ממויינים בצורת לוח של עמודות ושורות שיש יחס ביניהם, כדוגמת לוח הכפל (טבלה דו-ממדית).

טבלאות יכולות להיות באורך קבוע, כמוסבר בפרק זה, או באורך משתנה. אפשרויות נוספות לניהול טבלה ראה בספרות המתאימה.

דוגמה לטבלה:

שם הטבלה: מחירי פריטים

איבר 1	שם פריט1	מחיר 1
איבר 2	שם פריט2	מחיר 2
איבר 3	שם פריט3	מחיר 3
-----	---------	------
-----	---------	------

טבלה זו היא במבנה רשימה (חד-ממדית). כל איבר מורכב משני שדות: שם פריט ומחיר. כל האיברים זהים במבנה שלהם ושונים בתוכן. שים לב שאיברי הטבלה צריכים להיות מוגדרים ברציפות באחסנת המחשב.

הגדרת איברי הטבלה - OCCURS

א. מופע ומבנה איברי הטבלה

טבלה מגדירים ב-WORKING-STORAGE באופן דומה להגדרת רשומות. שם הטבלה מוגדר באזור A ברמת שדה 01. פרוט האיברים ניתן באזור B, ברמות שדה גבוהות יותר. מכיון שכל איברי הטבלה זהים במבנה, אפשר להגדיר במפורט את המבנה של אחד מהם ולציין את מספר הפעמים שהוא חוזר על עצמו. האיבר של הטבלה יכול להיות בעל שדה אחד, בעל מספר שדות ברמת שדה שוה אי כמה שדות ברמות שדה שונות.

מספר הפעמים שאיבר הטבלה חוזר על עצמו בטבלה מצויין ע"י מלת המפתח OCCURS המופיעה במשפט ההגדרה של השדה. אסור לציין OCCURS ברמת שדה 01. להלן הגדרת איברי טבלה עם שדה אחד:

L_1 (שם הטבלה).

L_1 (שם שדה) OCCURS n PIC (מבנה השדה).

L_1 - רמת השדה של הטבלה,

(שם הטבלה) - כשמו מאפשר התיחסות אל איברי הטבלה כמכלול אחד,

L_2 - רמת השדה של איברי הטבלה,

(שם שדה) - כשמו. שם השדה של איבר הטבלה,

n - מספר הפעמים שהשדה (האיבר) מופיע בטבלה,

PIC - יש לתאר את מבנה השדה.

דוגמאות:

1) טבלה שבה כל איבר מורכב משדה אחד בלבד

```
01  TABLE.
    03 STUDENT ØCCURS 30 PIC A(20).
```

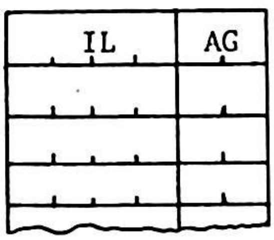
שם הטבלה רשום ברמת-שדה 01 ופרוט השדות ברמת-שדה 03. שם השדה של איבר הטבלה הוא STUDENT והוא מופיע בטבלה 30 פעם. המבנה - A(20). מכאן שאורך הטבלה הוא 20 × 30 = 600 תוים.

2) טבלה שבה כל איבר מורכב משני שדות:

```
01 LIST.
   03 MØNEY ØCCURS 20.
      05 IL PIC 9(4).
      05 AG PIC 99.
```

שם הטבלה הוגדר ברמת-שדה 01. בטבלה 20 איברים המוגדרים כנתון קבוצתי MONEY. MONEY מורכב משני שדות AG ,IL לציון לירות ואגורות. שים לב שמבנה השדה ניתן ברמת-השדה הגבוהה ביותר (כלומר המפורטת ביותר), כמקובל.

- שם הטבלה: LIST
- שם כל איבר: MONEY
- בכל איבר שני שדות: AG ,IL
- אורך כל איבר 6 תוים
- אורך כל הטבלה (20×6) 120 תוים



20 איברים בטבלה

טבלה זו הינה חד-ממדית, כדוגמת רשימה, באורך קבוע.

ב. <u>אינדקס הטבלה</u>

שמות השדות זהים לכל האיברים של הטבלה ולכן אם מוזכר שם כזה בהוראה, המהדיר לא ידע לאיזה מהאיברים להתיחס. לכן, בכל התיחסות לשדה (או לשם האיבר) בטבלה יש לציין לאיזה מאיברי הטבלה הכוונה. הציון ייעשה ע"י אינדקס בסוגריים כדלקמן: (אינדקס מספרי) שם-שדה

ולדוגמה. ... MØVE NAME (3) TØ

ההתיחסות במשפט זה היא לשדה ששמו NAME הנמצא באיבר השלישי של הטבלה. שים לב שהפניה היא לשם האיבר בטבלה ולא לשם הטבלה.

צורת כתיבה זו אינה נוחה תמיד כי ההוראה תתיחס תמיד לאיבר השלישי. בתכנות דרושה פנייה דינמית לאיברי הטבלה כדי לאפשר מעבר על הטבלה בלולאה (loop), כלומר להשתמש באותן הפקודות כדי לטפל בכל האיברים. לשם כך משתמשים באינדקס משתנה במקום אינדקס שהוא מספר קבוע. האינדקס מוגדר כשדה מספרי אשר מקבל ערכים שונים במהלך ביצוע התוכנית. ההתיחסות לשדה זה כאינדקס מכוונת לערך שיש בו בזמן הביצוע של ההוראה.

<u>דוגמאות:</u>

1)

```
COMPUTE NUMBER = A + 2.
MØVE NAME (NUMBER) TØ PRINT-AREA.
```

שדה NUMBER מחושב בהתאם להוראות הבעיה. כאשר הוא משמש כאינדקס תהיה פניה אל איבר הטבלה שמספרו שוה לערך של NUMBER בעת ביצוע ההוראה MOVE.

2) יש לבנות טבלה של מספרים זוגיים מ-0 עד 100.

הפתרון:

```
77  VAR PIC 9(4).
77  MØNE PIC 9(3) VALUE 0.
01  EVEN-NUMBERS.
      03 GØREM ØCCURS 50 PIC 9(3)
---------
LOOP. PERFØRM MILUI-TAVLA VARYING
             VAR FRØM 1
             BY 1 UNTIL VAR > 50.
         STØP RUN.
MILUI-TAVLA  ADD 2 TØ MØNE.
      MØVE MØNE TØ GØREM (VAR)
```

<u>הסבר:</u>

האינדקס VAR (שורה 1) מוגדר כנתון אלמנטרי מספרי. שדה MONE מיועד לחי-שוב הערך של המספר הזוגי שיש להעביר לטבלה.

הטבלה מוגדרת ברמת שדה 01 (שורה 3). מכיון שאי אפשר לציין OCCURS ומבנה שדה ברמה זו, יש להגדיר נתון אלמנטרי GOREM במבנה של 3 ספרות אשר מופיע 50 פעם בטבלה (שורה 4).

התוכנית (שורה 5) כוללת הוראת PERFORM מורכבת לביצוע השגרה הכלולה בקטע התוכנית MILUI-TAVLA (שורות 9-10). הוראה זו תפעיל את השגרה מספר פעמים כאשר האינדקס VAR משתנה מ-1 בקידום של יחידה עד אשר יהיה גדול מ-50. כלומר כאשר יגיע ל-51 יופסק המחזור, והתוכנית לא תעבור לביצוע השגרה אלא תעבור להוראה העוקבת להוראת PERFORM, אל ההוראה STOP RUN (שורה 8). זכור שקידום האינדקס נעשה לאחר ביצוע השגרה.

בשגרה (שורה 9) מוסיפים 2 אל השדה MONE (אשר ערכו ההתחלתי 0, לפי שורה 2) ומעבירים אותו אל האיבר הסידורי בטבלה בהתאם לערכו של האינדקס VAR בעת ביצוע ההוראה. בפעם הראשונה VAR=1 ולכן מתכוונים לאיבר הראשון GOREM(1), אח"כ ערכו מקודם ל-2 ואז איבר הטבלה הוא GOREM(2) וכן הלאה.

תרגיל מסכם לשגרות וטבלאות

מטרת התרגיל להציג שימוש בטבלאות ובהוראת PERFORM.

<u>תאור הבעיה</u>

נתונות שתי כרטסות של חברי מיעדון ספורט. כרטסת אחת כוללת את רשימת החברים, והכרטסת השניה - את רשימת החברים ששילמו את מיסיהם השנתיים. יש לבדוק כמה חברים טרם שילמו את מיסיהם ולהרפיס מספר זה בסיום העבודה. כאשר ישנו חבר ששילם את מיסיו ואיננו מופיע ברשימת החברים, או כאשר יש לחבר דווח כפול על תשלום מס אין להתיחס לכך, אלא לדלג על הכרטיס (הסיבה - לפשט את הבעיה. כך <u>אין</u> נוהגים במציאות).

הקלט: א. כרטסת חברי המועדון במבנה הבא:
טור 1 : קוד כרטיס (1).
טורים 9-2: מספר זהות.
שאר טורי הכרטיס לא רלוונטיים בתרגיל זה.

כרטסת חברי המועדון ממויינת לפי מספר תעודת זהות, ויש בה עד 200 כרטיסים.

ב. כרטסת חברים ששלמו מסיהם במבנה הבא:
טור 1 : קוד כרטיס (2).
טורים 9-2: מס' זהות.
שאר טורי הכרטיס לא רלוונטיים בתרגיל זה.

כרטסת משלמי המיסים אינה ממויינת.

מזינים תחילה את כרטסת החברים ולאחריה - כרטסת משלמי המיסים. בכרטיס האחרון של שני קבצי הנתונים מנוקבת כוכבית (*) בטור 1 ואין בו נתונים.

הצגת דרך הפתרון:

בשלב זה ננתח את השלבים העיקריים בפתרון הבעיה:

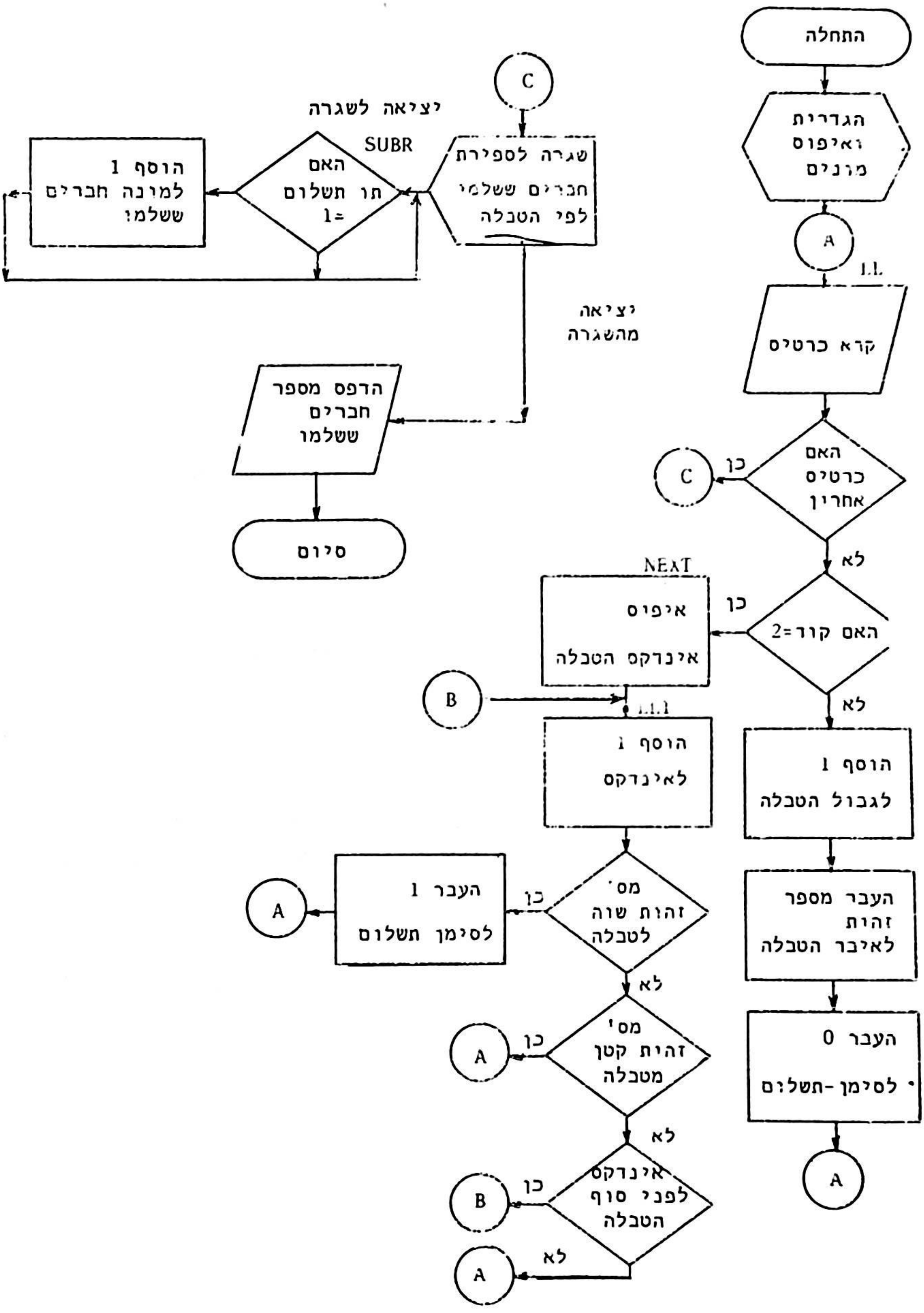
א. הגדרת טבלה של 200 איברים, בהם נאחסן את מספרי תעודות הזהות של החברים במועדון. נציין תו נוסף בכל איבר לציון עובדת התשלום כאשר נמצא שחבר זה שילם מסיו.

ב. קריאת כרטסת החברים ומילוי אברי הטבלה עם מספרי תעודת הזהות.

ג. קריאת כרטסת משלמי המסים, חיפוש בטבלה, וציון עובדת התשלום כאשר מוצאים חבר עם מספר ת"ז מקביל בטבלה, או דילוג על הכרטיס (זהו בדרך כלל מצב של שגיאה).

ד. סריקת הטבלה לספירת החברים שטרם שילמו מסיהם, והדפסת מספר זה במכונת הכתיבה של המפעיל.

תרשים מלבני:


התחלה
הגדרית ואיפוס מונים
A
I.I.
קרא כרטיס
האם כרטיס אחרון
כן
C
לא
האם קוד=2
כן
NEXT
איפוס אינדקס הטבלה
לא
הוסף 1 לגבול הטבלה
העבר מספר זהות לאיבר הטבלה
העבר 0 לסימן-תשלום
A
B
I.I.I
הוסף 1 לאינדקס
מס' זהות שוה לטבלה
כן
העבר 1 לסימן תשלום
A
לא
מס' זהות קטן מטבלה
כן
A
לא
אינדקס לפני סוף הטבלה
כן
B
לא
A
C
שגרה לספירת חברים ששלמו לפי הטבלה
יציאה לשגרה
SUBR
האם תו תשלום =1
הוסף 1 למונה חברים ששלמו
יציאה מהשגרה
הדפס מספר חברים ששלמו
סיום

<u>תוכנית קובול</u>

```
   A   B
   IDENTIFICATIØN DIVISIØN.
   PRØGRAM-ID. 'TARGIL'.
   ENVIRØNMENT DIVISIØN.
   DATA DIVISIØN.
   WØRKING-STØRAGE SECTIØN.

   77  MØNE PIC 9(3) VALUE 0.
   77  RATZ PIC 9(3) VALUE 0.
   77  GVUL PIC 9(3) VALUE 0.

   01  KARTIS.
       02 KØD PIC X.
       02 ZEHUT PIC 9(8).
       02 FILLER PIC X(71).

   01  TAVLA.
         03 GØREM ØCCURS 100.
             05 ZEHUT-T PIC 9(8).
             05 SIMAN-T PIC X.

   PRØCEDURE DIVISIØN.

   LL. ACCEPT KARTIS. IF KØD = '*' GØ TØ SFØR.
              IF KØD = '2' GØ TØ NEXT.
              ADD 1 TØ GVUL. MØVE ZEHUT TØ ZEHUT-T (GVUL).
               MØVE '0' TØ SIMAN-T (GVUL). GØ TØ LL.
   NEXT. MØVE ZERØ TØ RATZ.
   LL1. ADD 1 TØ RATZ. IF ZEHUT = ZEHUT-T (RATZ)
         MØVE '1' TØ SIMAN-T (RATZ) GØ TØ LL.
        IF ZEHUT < ZEHUT-T (RATZ) GØ TØ LL
        IF RATZ < GVUL GØ TØ LL1. GØ TØ LL.
   SFØR. PERFØRM SUBR  VARYING RATZ FRØM 1
                                BY 1
                                UNTIL RATZ > GVUL.
         DISPLAY MØNE UPØN CØNSØLE.
         STØP RUN.
   SUBR. IF SIMAN-T (RATZ) = '1' ADD 1 TØ MØNE.
```

הערות לפתרון התרגיל

א. הגדרות של שדות ברמה 77 קודמות להגדרות של שדות ברמות האחרות. (שורות 8-6).

ב. כל איבר בטבלה מכיל שני שדות, ולכן צריך להגדירו כנתון קבוצתי. מכיון שרמה 01 אסור לציין את המלה OCCURS צריכים להגדיר נתון קבוצתי לצורך זה - GOREM ובתוכו להגדיר את שני הנתונים המרכיבים אותו. (שורות 13 עד 16).

ג. הסבר ללוגיקה של התוכנית:

1) לאחר קריאת הכרטיס בודקים אם זהו הכרטיס האחרון (שורה 18). אם כן - מסתעפים לקטע SFOR (בשורה 27) שמבצע ספירת החברים שטרם שילמו. אם קוד הכרטיס = 1 (כלומר אינו = 2) זהו כרטיס חבר שאת תוכנו יש להעביר לטבלה (שורה 19). האינדקס (GVUL) שמצביע על האיבר הסידורי בטבלה, מקדמים אותו ב-1 (מכיון שהתחיל מ-0) ומעבירים את מספר הזהות שבכרטיס לשדה המתאים של הטבלה (שורות 21-20). את הסימן לתשלום השייך לאיבר זה מאפסים כי טרם קבלנו הודעת תשלום עבור התושב הנדון. חוזרים לקריאת כרטיס חדש. בסיום קריאת כרטיסים "קוד 1", יכיל GVUL את מספר האיברים בטבלה.

2) אם קוד הכרטיס 2, חייבים לסרוק את כל הטבלה מהאיבר הראשון (1) ועד האיבר האחרון שמולא ע"י הכרטיסים בקוד 1. מספר איבר זה נשמר בשדה GVUL (שורות 26-22).
הסריקה נעשית בעזרת האינדקס RATZ המשתנה מ-1 עד הערך GVUL (שורות 27-23). אם נמצא שמס' תעודת זהות בכרטסת החברים ששילמו מתאימה למספר ת"ז בטבלה מסמנים באותו איבר את עובדת תשלום המיסים. אם לא נמצא, כלומר הסריקה עברה כל הטבלה בלי למצוא תעודת זהות מתאימה, מדלגים לכרטיס הבא (כאמור - זהו מצב של שגיאה אשר בעבודה רגילה מקובל לדווח עליו).

3) עם גמר קריאת כל הכרטיסים מסתעפים ל-SFOR שם סורקים את כל הטבלה תוך ספירת התושבים ששילמו את מיסיהם (שורות 27 עד 29, 32). מספר זה יש להדפיס בסוף העבודה (שורות 30, 31) ולאחר מכן - לסיים את התוכנית. סריקת הטבלה נעשית ע"י שגרה SUBR (שורה 32).

פרק 6: קלט ופלט בקבצים

עד כה הכרנו את הוראות קלט/פלט ACCEPT ו-DISPLAY. אלה הן הוראות פשוטות שיכולות לשמש במהלך תוכנית לקריאת כרטיסים בודדים או להדפסת שורות מעטות במיוחד אמורים הדברים לגבי מכונת הכתיבה האיטית. כשעוסקים בקבצי כרטיסים גדולים ובדוחות מירכבים עם עריכה (כותרת בראש דף, רווחים וכדי) יש צורך להשתמש בהוראות קלט/פלט המאפשרות טיפול בכל יחידות קלט/פלט כולל אמצעי קלט/פלט מגנטיים כמו סרטים ותקליטים. חשוב יותר הוא השרות לבדיקת תויות כותרת וסיומת של קבצים, ביצוע שרותי קידום נייר במדפסת, הפרדת כרטיסים בקוראת הכרטיסים ועוד.

היחידה הבסיסית בקלט/פלט היא הקובץ. כדי לשתף קובץ בתוכנית יש לבצע שתי פעולות: הגדרת הקובץ וכתיבת הוראות לעיבודו.

הגדרת קבצים

הגדרת קבצים נעשית בחלקה ב-ENVIRONMENT DIVISION ובחלקה ב-DATA DIVISION.

<u>הכרזת קובץ ויחידת קלט/פלט - SELECT</u>

משפט זה קובע <u>שם</u> לקובץ ואת הקשר הלוגי בינו לבין יחידת קלט/פלט.
המשפט ייכתב ב-ENVIRONMENT DIVISION,ב-INPUT-OUTPUT SECTION, תחת הכותרת FILE-CONTROL.

```
A   B
IDENTIFICATIØN DIVISIØN.
PRØGRAM-ID. 'TARGIL'.

ENVIRØNMENT DIVISIØN.
INPUT-ØUTPUT SECTIØN.
FILE-CØNTRØL.
    SELECT----
    SELECT----
```

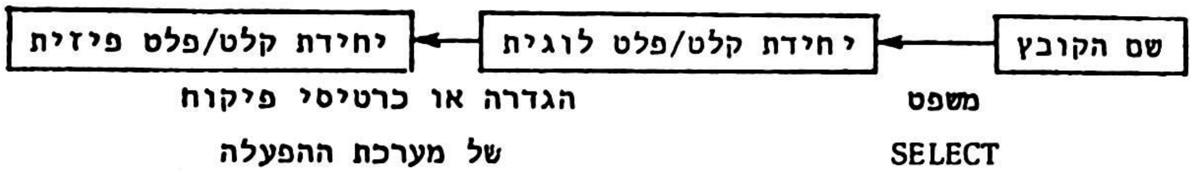
בכל תוכנית חייב להופיע משפט SELECT אחד עבור <u>כל</u> קובץ. מבנה המשפט הוא כדלקמן:

SELECT (שם קובץ) ASSIGN TØ (שם יחידת קלט/פלט)

"שם הקובץ" הוא שם לוגי שנקבע ע"י המתכנת וישמש לזיהוי הקובץ בכל הפניות אליו בתוך התוכנית.

"שם יחידת ק/פ" הוא כינוי לוגי ליחידת קלט/פלט (למשל כונן סרט מגנטי, כונן דיסק מגנטי, מדפסת וכד') שאליה קשור הקובץ. הכינוי הלוגי יכול להתיחס ליחידה פיזית לפי הגדרה קבועה במערכת ההפעלה, או שאפשר לשנות הגדרה זו ע"י כרטיסי פיקוח מתאימים. בכל מקרה, הכינויים וההגדרות שונים בדרך כלל ממתקן מחשב אחד למשנהו.

להלן תאור הקשר בין שם הקובץ ליחידת קלט/פלט:



למשל: קובץ "מלאי", קשור ליחידת ק/פ a, שהיא כונן סרטים n.

כדוגמה יוצגו מספר משפטים המתאימים למערכות י.ב.מ./370. אך יש לזכור שמשפט SELECT מותנה בסוג המחשב ובמערכת ההפעלה שבהם פועלת התוכנית ויש לכתוב אותו בהתאם להנחיות בכל מערכת.

- קריאת קובץ כרטיסים:

SELECT NEW-CARDS ASSIGN TØ UR-2540R-S-CARDIN.

או בקיצור: SELECT NEW-CARDS ASSIGN TØ CARDIN.

UR-2540R מציין את סוג יחידת הקלט/פלט ואת השם הפיזי של קורא הכרטיסים.
S מציין את שיטת עיבוד הקובץ, כאן - קובץ סיקוונציאלי,
CARDIN - כינוי לוגי של יחידת קלט/פלט.

- קריאה/כתיבה של קובץ בסרט מגנטי:

SELECT MASTER-FILE ASSIGN TØ TAPEA.

- כתיבת דוח (קובץ) במדפסת:

SELECT LIST-ØF-ITEMS ASSIGN TØ SYSPRNT.

אפשר להשמיט את התאור המפורט של יחידת קלט/פלט ולציין <u>רק</u> את הכינוי של היחידה הלוגית אשר מוכר למערכת ההפעלה מכיון ששאר תכונות היחידה ידועות לה.

הגדרת תכונות הקובץ - משפט FD

הגדרת תכונות הקובץ נעשית ב-DATA DIVISION. משפט FD (File Description) נכתב ב-FILE SECTION המופיע לפני ה-WORKING-STORAGE SECTION שהכרנו עד כה.

לכל קובץ שמוגדר ע"י משפטSELECT יש לכתוב משפט FD מתאים. במשפט FD מספר מלות מפתח שחלק מהן חייבות להופיע וחלק מהן יכולות להופיע. הנקודה המסיימת את משפט ה-FD תופיע לאחר הפסוק האחרון.

```
            DATA DIVISIØN.
            FILE SECTIØN.
חובה 1      FD (שם קובץ)
חובה 2          LABEL RECØRD IS {STANDARD / ØMITTED},
רשות 3          RECØRDING MØDE IS (קוד),
רשות 4          BLØCK CØNTAINS (מספר) {CHARACTERS / RECØRDS},
חובה 5          RECØRD CØNTAINS (מספר) CHARACTERS,
חובה 6          DATA RECØRD IS (שם רשומה).
```

הסבר למבנה ההוראה:

שורה 1: השם FD נכתב באזור A. שאר מלות המפתח נכתבות במבנה חופשי באזור B. (שם הקובץ) הוא אותו השם שנקבע במשפט ה-SELECT ב-ENVIRONMENT DIVISION.

שורה 2 (חובה): .. LABEL - פסוק מגדיר את התוית של הקובץ. האפשרויות הן:
STANDARD - תוית סטנדרטית
OMITTED - ללא תוית כלל.

האופציה OMITTED חייבת להופיע לגבי קבצים המתיחסים לקוראת כרטיסים ומדפסות, ויכולה להופיע לגבי קבצים המתיחסים לסרטים מגנטיים. קבצים על גבי דיסקים מגנטיים חייבים להיות עם תוית סטנדרטית.

שורה 3 (רשות): ... RECORDING

פסוק קובע מבנה הרשומות בקובץ:
ה(קוד) יכול לקבל אחד מהערכים הבאים:

F - הרשומות הן באורך קבוע (Fixed),
V - הרשומות הן באורך משתנה (Variable),
U - הרשומות במבנה לא מוגדר (Undefined).

אם פסוק זה אינו מופיע לגבי קובץ מסויים - התוכנית מניחה רשומות מסוג V.

<u>שורה 4</u> (רשות): פסוק ... BLOCK קובע את אורך הגוש (block) של הקובץ. אורך הגוש נקבע ע"י מספר הרשומות שבו (n RECORDS) או ע"י מספר התוים שבו (n CHARACTERS) הפסוק לא יופיע כשהקובץ הוא במבנה unblocked, כלומר רשומה אחת לבלוק וכאשר הקבצים הם במבנה רשומות בלתי מוגדר (undefined). פסוק זה הוא חובה לקובץ הערוך בבלוקים.

<u>שורה 5</u> (חובה): הפסוק ... RECORD קובע את אורך הרשומה בקובץ. כאשר הרשומות הן באורך קבוע - זהו אורך הרשומה. כאשר הרשומות הן באורך משתנה או לא מוגדר - זהו האורך המקסימלי האפשרי.

<u>שורה 6</u> (חובה): הפסוק ... DATA מציין את <u>שם הרשומה</u> אשר <u>תוגדר מיד בהמשך</u> למשפט FD. זוהי רשומת קלט <u>לאחר</u> קריאתה (בקבצי קלט), או רשומת הפלט <u>לפני</u> כתיבתה (בקובצי פלט). בהגדרה זו מתיחסים לרשומה הבודדת. כאשר התוכנית קוראת בלוקים (למשל מסרט מגנטי) היא "מפרקת" אותם ומעבירה לשטח הרשומה את הרשומה הבאה בתור. כאשר כותבים בלוקים, "מלקטת" התוכנית את הרשומות לבלוקים ורק כאשר הבלוק מושלם הוא נכתב למעשה ביחידת הפלט. כרטיסים ושורות הדפסה קוראים וכותבים ברשומות בודדות.

<u>גדרת רשומת קובץ:</u>

להגדרת רשומה הקשורה לקובץ יש מספר כללים:

א. הרשומה חייבת להיות ברמה 01, ויכולה להכיל פרוט לרמות גבוהות יותר, אם נדרש.
ב. אי אפשר להשתמש בהגדרה חוזרת של שטח הרשומה ע"י משפט REDEFINES.
ג. אי אפשר להשתמש בהגדרת VALUE בתאור המבנה ע"י PICTURE. (הגדרת VALUE דרושה בעיקר להצגת קבועים ברשומות להדפסה).

מסיבות אלו נוהגים לפעמים לטפל ברשומות ב-WORKING STORAGE ולהגדיר את רשומות הקובץ כשדה אחד במבנה אלפנומרי, למשל:

01 (שם רשומה) PIC X(n).

<u>תרגיל הגדרת קבצים</u>

יש לכתוב את חלק התוכנית המגדיר את שני הקבצים הבאים:

1) קובץ על סרט מגנטי, תוית סטנדרטית, אורך רשומה 40 תוים ואורך גוש 400.

2) קובץ כרטיסים מנוקבים.

```
       A   B

 1     IDENTIFICATIØN DIVISIØN.
 2     PRØGRAM-ID. 'KVATZIM'.
 3     ENVIRØNMENT DIVISIØN.
 4     INPUT-ØUTPUT SECTIØN.
 5     FILE-CØNTRØL.
 6         SELECT SERET ASSIGN TØ SYS003.
 7         SELECT CARTIS ASSIGN TØ SYSIPT.
 8     DATA DIVISIØN.
 9     FILE SECTIØN.
10     FD. SERET  LABEL RECØRD IS STANDARD,
11                RECØRDING MØDE IS F,
12                BLØCK CØNTAINS 10 RECØRDS,
13                RECØRD CØNTAINS 40 CHARACTERS,
14                DATA RECØRD IS SERET-AREA.

15     01   SERET-AREA.
16             02 KØD PIC 9.
17             02 NAME PIC X(19).
18             02 ADDRESS PIC X(20).

19     FD   CARTIS LABEL RECØRD IS ØMITTED
20                 RECØRDING MØDE IS F
21                 RECØRD CØNTAINS 80 CHARACTERS
22                 DATA RECØRD IS CARTIS-AREA.

23     01   CARTIS-AREA PIC X(80).
```

<u>הסבר:</u>

א. בשורות 7-5 מוגדרים הקבצים: קובץ SERET אשר יופעל ביחידה SYS003 (המציינת כונן סרט מגנטי) וקובץ CARTIS אשר יקרא מיחידה SYSIPT.

ב. בשורה 10 מוגדר הקובץ בסרט מגנטי, בהתאם לדרישות הבעיה. אורך הבלוק (שורה 12) צוין ככפולה של מספר הרשומות ולא כאורך פיזי. כאשר הקובץ בנוי מרשומות באורך משתנה או במבנה לא מוגדר מציינים את אורכו המקסימלי ע"י מספר התוים ולא ע"י מספר הרשומות. בשורה 14 מוגדר שם הרשומה המיועדת לקריאה/כתיבה בסרט.

ג. רשומת הסרט מפורטת בשורות 15 עד 18. שים לב שאורך הרשומה הוא 40 תוים כפי שצויין בשורה 13.

ד. קובץ הכרטיסים מוגדר בשורות 19 עד 22. לקובץ זה אין תוית ואין מבנה בלוקים. שם הרשומה ניתן בשורה 22.

ה. רשומת הקלט של הכרטיסים מפורטת בשורה 23. לרשומה שדה אחד בלבד בן 80 טורים.

הוראות לעיבוד קבצים

פתיחת קבצים - OPEN

הוראת OPEN היא ההוראה הראשונה שחייבת להתבצע עבור כל קובץ, ואין להתחיל בקריאה או כתיבה ללא הוראה זו. ההוראה גורמת להכנת הקובץ לעיבוד.

בקבצי קלט ופלט בכרטיסים ובקובץ פלט במדפסת ההוראה אינה מבצעת דבר. בקובץ קלט בסרט מגנטי מבוצע גילגול הסרט לתחילתו, והתוית נקראת ונבדקת. בקובץ בדיסק מתבצעת בדיקת התוית. בקובץ פלט בסרט מגנטי ובדיסק נכתבת (או מתעדכנת) התוית של הקובץ, והקובץ מוצב במקום הפיזי המיועד לכתיבת הבלוק הראשון.

מבנה ההוראה:

ØPEN { INPUT / ØUTPUT } (שם קובץ).

(שם קובץ) הוא שם הקובץ שצויין במשפט SELECT ובמשפט FD. ההגדרה של סוג הקובץ (קלט או פלט) מיועדת לבדיקת ההרשאה לביצוע פעולות קריאה או כתיבה.

אפשר לציין מספר קבצים בהוראה אחת, ואף קבצים מסוגים שונים, אך בכל רשימה יש לכלול רק קבצים מאותו הסוג, לדוגמה:

```
ØPEN INPUT KØVETZ1 MASTER ØUTPUT DØCH DISK-A.
```

הוראת קריאה - READ

קריאת רשומות מקובץ נעשית ע"י ההוראה READ:

READ (שם קובץ) AT END (משפט קובול).

הוראה זו תקרא רשומה מהקובץ ששמו צויין בהוראה. לאחר ביצוע ההוראה יתכן אחד משני מקרים:

א) היו רשומות בקובץ - הרשומה שנקראה נמצאת בזכרון בשטח שצויין לאותו קובץ במשפט ה-FD (----- DATA RECORD IS), ואשר הוגדר מיד בהמשך למשפט ה-FD המתאים לקובץ.

ב) לא היו יותר רשומות בקובץ - כאשר מגיעים לסוף הקובץ מתבצע משפט הקובול שצויין בהוראה בהמשך ל- ... AT END. חובה לרשום ... AT AND בכל הוראות READ או לא לרשום כלל. אם לא רושמים- חובת בדיקת סוף הקובץ מוטלת על התוכניתן.

דוגמה: READ INPUT3 AT END STØP RUN.

קריאה של רשומות מקובץ INPUT3. כאשר מגיעים לסוף הקובץ מסתיימת התוכנית.

READ TRANSACTIØNS AT END GØ TØ FINISH.

קריאה של רשומות מקובץ TRANSACTIONS. בסיום קריאת הקובץ יבוצע סיעוף לקטע התוכנית FINISH.

הסבר לאופן הקריאה של הרשומות:

כאשר קריאת הרשומות מתבצעת מקובץ שהרשומות בו נפרדות (unblocked) מתבצעת פעולת קריאה פיזית בכל הוראת READ כמו למשל בקריאת כרטיסים מנוקבים.

כאשר קריאת הרשומות מתבצעת מקובץ שהרשומות בו מקובצות בבלוקים (blocked) נוהל הקריאה הוא: בהוראת הקריאה הראשונה מתבצעת ע"י קובול פעולת קריאה פיזית אל תוך שטח שמוגדר אוטומטית ואינו מוכר למשתמש. לאחר מכן נעשית פעולת קריאה לוגית שהיא העברת רשומה אחת מהבלוק אל שטח הזכרון שהוגדר ע"י המשתמש. בפעולת הקריאה הבאה תבוצע שוב פעולת קריאה לוגית. כאשר הסתיימה קריאת כל הרשומות בבלוק, תבוצע שוב פעולת קריאה פיזית ולאחריה פעולה לוגית וחוזר חלילה, עד לסיום הקובץ.

קריאת רשומה לשטח אחסנה אחר (INTO):

הוראת READ תגרום תמיד לקריאת הרשומה אל הרשומה של הקובץ המוגדרת לאחר משפט FD. כאשר צריך להעביר את הרשומה לעיבוד בשטח נתונים אחר (ב-WORKING STORAGE) או לשטח רשומת פלט כלשהיא, משתמשים בהוראה MOVE.

אם הוגדר לדוגמה: FD TALMIDIM ----- .

01 R-TALMID PIC -----.

WØRKING-STØRAGE SECTIØN.

01 NEW-AREA PIC -----.

כותבים: READ TALMIDIM.

MØVE R-TALMID TØ NEW-AREA.

ניתן להשתמש בברירה INTO במשפט READ כדי לבצע שתי פעולות אלו במשפט אחד:

READ (שם קובץ) [INTØ] (שטח אחסנה) AT END (משפט קובול).

ולדוגמה: READ TALMIDIM INTØ NEW-AREA
AT END STØP RUN.

הוראת כתיבה - WRITE

כתיבת רשומות בקובץ (כולל הדפסה) נעשית ע"י ההוראה WRITE.

WRITE (שם רשומה).

בהוראת WRITE אין מציינים שם הקובץ אלא רק את שם הרשומה אשר צויין במשפט FD (במלת המפתח ... DATA RECORD IS) של הקובץ. תאור הרשומה ניתן בהמשך למשפט FD.

לדוגמה: WRITE ØUT-AREA.

הרשומה שנמצאת בעת ביצוע ההוראה בשטח OUT-AREA תכתב בקובץ אשר OUT-AREA הוא שטח הרשומות שלו. מכאן יש להסיק שהמתכנת צריך להכין את הנתונים לפלט ברשומה שהוגדרה כרשימת פלט.

הסבר לאופן הביצוע של הוראת הכתיבה:

כשם שהוראת הקריאה אינה מבצעת תמיד קריאה פיזית, כך הדבר גם בהוראת הכתיבה. כאשר כותבים רשומות לקובץ שהרשומות בו נפרדות (unblocked) כמו למשל במדפסת, תבצע כל הוראת כתיבה פעולה פיזית של כתיבה בקובץ.
כאשר כותבים רשומות לקובץ שהרשומות בו מקובצות בבלוקים תבוצע פעולת כתיבה לוגית עד אשר "יתמלא" הבלוק, ורק אז תבוצע פעולת כתיבה פיזית. הכתיבה הפיזית

"שקופה" למשתמש והוא אינו מטפל בה, כשם שאין הוא יודע מתי מתבצעת פעולת קריאה פיזית. הבלוק המיועד לצבירת הרשומות נשמר בשטח הקובול מחוץ לשליטת המשתמש. בכתיבה קיימת בעיית הבלוק האחרון אשר עשוי להיות מלא בחלקו ועליו להכתב עם סיום התוכנית. פעולה זו מבוקרת בעזרת ההוראה CLOSE כפי שיוסבר.

הוראת כתיבה משטח אחסנה אחר (FRØM)

קיימים מצבים בהם הרשומה המיועדת לפלט נמצאת בשטח אחסנה כלשהוא לאחר עיבוד, עריכה או כרשומה קבועה (למשל כותרת להדפסה), או שהיא נמצאת בשטח רשומת קלט. לצורך ביצוע ההוראה WRITE יש להעביר תחילה את הרשומה הרצויה אל שטח רשומת קובץ הפלט. כדי להקל על התכנות אפשר לבצע שתי פעולות אלו במשפט אחד:

WRITE (רשומת קובץ פלט) FRØM (שטח אחסנה).

לדוגמה, במקום לכתוב:

```
MØVE RESHUMA-A TØ SHURA.
WRITE SHURA.
```

אפשר לכתוב:

```
WRITE SHURA FRØM RESHUMA-A.
```

שיטת עבודה זו דרושה במיוחד בקבצים המיועדים להדפסה מכיון שבהם יש צורך להדפיס שורות במבנה שונה אשר עורכים אותן ב-WORKING-STORAGE, או שורות כותרת והסברים שונים אשר מוגדרות גם כן בפרק זה של חטיבת הנתונים.

כאשר משתמשים בשיטת עבודה זו לגבי קובץ מסויים, צריך להגדיר את מבנה רשומת הקובץ בצורה הבאה:

```
FD  PELET _____
01  RESHUMAT-PELET PIC X(n).
```

כאשר n היא אורך הרשומה בתוים.

כללי העברה של ברירת FROM הם כללי העברה של שדה אלפנומרי: התוים מועברים החל מהצד השמאלי של השדה, אם השדה המקבל ארוך מהרשומה המועברת אליו הוא יתמלא בתוי רווח מימין, ואם הוא קצר מהרשומה המועברת - התוים העודפים יקוצצו.

הוראת כתיבה למדפסת

הכתיבה במדפסת כוללת שתי פעולות עיקריות: פיקוח על קידום הנייר והדפסת השורה.

במחשבי י.ב.מ. מקודדת הוראת הפיקוח בצורת תו ראשון בשורת ההדפסה,תו זה אינו מודפס. לאחר תו זה רשומים שאר התוים בשורה, כאורך שורת ההדפסה המותרת במדפסת. לדוגמה, שורת הדפסה המיועדת להדפסת 5 תוים ראשונים של האלפבית תוגדר באורך כולל של 6 תוים, בצורה הבאה:

```
01  PRINT-LINE.
    02 FILLER PIC X.
    02 DATA PIC X(5) VALUE 'ABCDE'.
```

הוראת הפיקוח לקידום הנייר ניתנת בהוראת ההדפסה WRITE והמהדיר, הוא אשר מתרגם אותה לתו-פיקוח ומעביר אותה למקום המתאים בשורת ההדפסה.

הוראת ההדפסה יכולה לקבל אחת מהצורות הבאות:

WRITE (שם רשומה) AFTER PØSITIØNING (מספר).
WRITE (שם רשומה) AFTER ADVANCING (מספר) LINES.
WRITE (שם רשומה) BEFØRE ADVANCING (מספר) LINES.

וראות אלו מכוונות לביצוע ההדפסה ולקידום הנייר במדפסת:

א. ההוראה AFTER POSITIONING תגרום לקידום הנייר <u>לפני</u> הדפסת השורה במספר השורות המצויין. תאפשרויות הן 1, 2, 3.
כאשר רושמים <u>0</u>, תבצע המדפסת קידום הנייר <u>לדף חדש לפני</u> הדפסת השורה. קידום הנייר לדף חדש מונחה ע"י "סרט מוביל" המותקן במדפסת. סימון השורה הראשונה הוא בדרך כלל בשורה השלישית מראש הדף.

ב. שתי ההוראות האחרות מאפשרות קידום הנייר במספר שורות (עד 100) כמצויין בהוראה. ההוראה המאפשרת הדפסת השורה <u>לפני</u> קידום הנייר היא יעילה יותר מבחינת קצב ההדפסה. כאשר מספר = 0 מבוצע קידום הנייר לדף חדש.
בכל מקרה שמציינים קידום של מספר שורות יהיה הרווח בין שתי השורות המוד-פסות קטן ב-1 ממספר השורות שצויינו בהוראה. למשל: אם מציינים מספר = 2, פרוש הדבר קידום הנייר ב-2 שורות, כלומר שורה אחת רווח ובשורה השניה תבוצע ההדפסה (זהו "רווח כפול").

איתור של סוף דף יכול להיות ע"י ספירת מספר השורות ובדיקת מספר זה לעומת קבוע

נתון. בדף הדפסה סטנדרטי יש 66 שורות, אך כמובן שמפסיקים את ההדפסה מספר שורות מתחתית הדף. קיימות אפשרויות נוספות לבדיקת מצב "סוף דף" אך לא נדון בהן במסגרת זו.

אם לא מציינים את ברירת קידום הנייר לקובץ המיועד למדפסת יוסיף המהדיר אוטומטית תו "קידום של שורה אחת". אך אם צויינה ברירה זו בהוראה - יש לרשום אותה בכל משפטי WRITE.

סגירת קבצים - CLOSE

בסיום העיבוד של כל קובץ יש לבצע את ההוראה CLOSE. הוראה זו משלימה את תהליך העיבוד בפעולות סיום אחדות, הכרחיות:

בקבצי קלט ופלט בכרטיסים ובקובץ פלט במדפסת לא נעשית כל פעולה. בקובץ קלט בסרט מגנטי או בדיסק מגנטי נעשית בדיקת התאמה של מספרי הבקורת בתוית הסיום של הקובץ אל המספר שחושב בזמן ביצוע הקריאה של הקובץ. במקרה של אי התאמה מודפסת הודעת שגיאה.

בקובץ פלט בסרט ובדיסק גורמת הוראת הסגירה לכתיבה פיזית של הבלוקים האחרונים הנמצאים בזכרון. במהלך העיבוד בלוקים אלה לא נכתבים עד אשר הם "מתמלאים" ברשומות, מה שלא קיים תמיד בסיום העיבוד. לאחר כתיבת הבלוקים האחרונים נכ-תבת תוית הסיום של הקובץ הכוללת מספרי בקורת שונים. בקבצים בסרטים מגנטיים מבוצע גם גלגול הסרט (בקלט או פלט).

מבנה ההוראה:

CLOSE (שם קובץ) (שם קובץ) ...

שים לב שאין צורך לציין את סוג הקובץ מכיון שעובדה זו ידועה לתוכנית ממשפט OPEN.

תרגיל מסכם לקלט ופלט

תאור הבעיה

נתון קובץ קלט בסרט מגנטי שמבנה רשומותיו הוא:

1- 8 מספר זהות

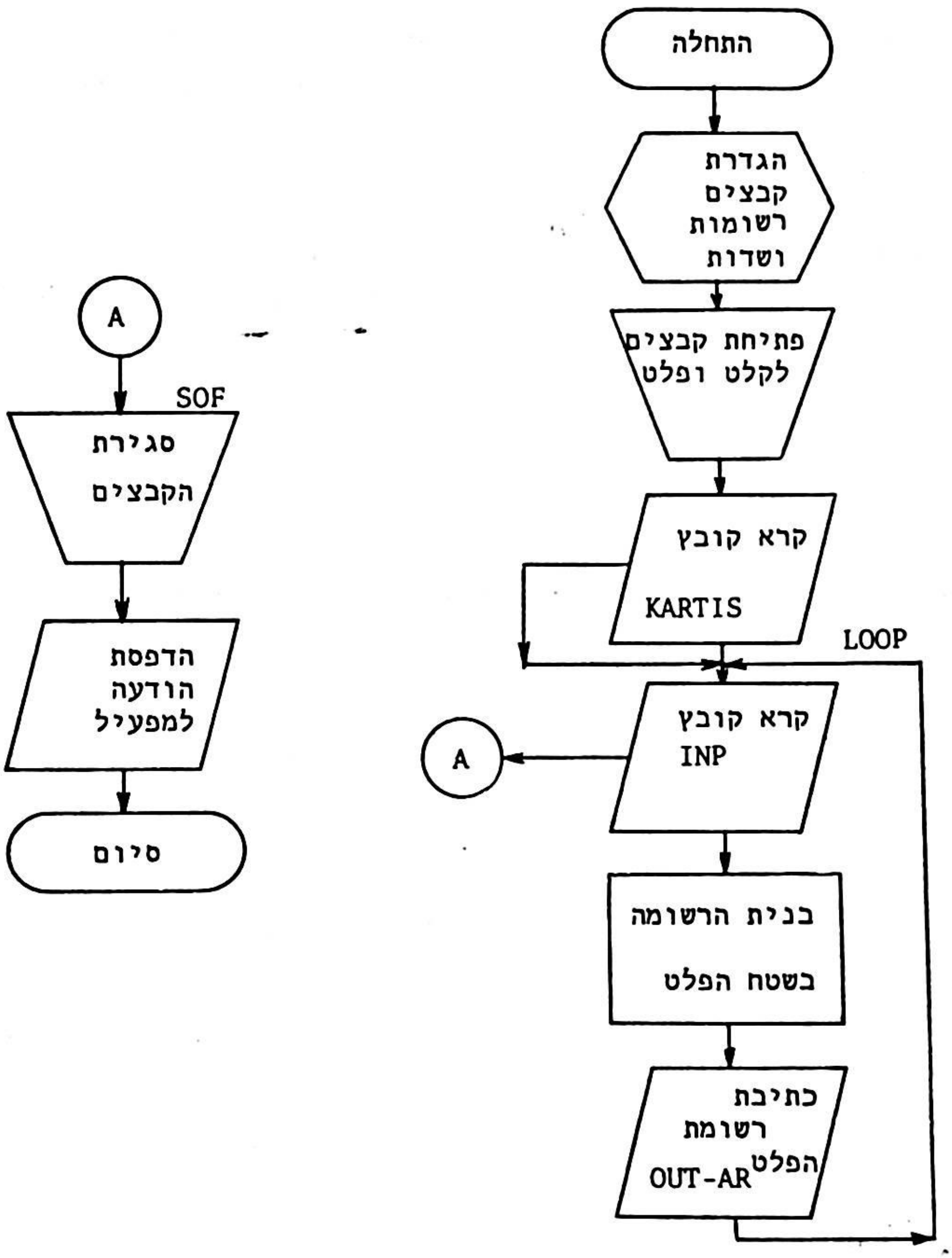
9-50 נתונים שונים

יש לכתוב קובץ פלט בסרט מגנטי אשר יכיל את הנתונים של רשומת הקלט ובנוסף, בכל רשומה, את המספר הסידורי שלה ואת תאריך העיבוד, במבנה הבא:

8 -1 מספר זהות
9-50 נתונים שונים
51-54 מספר סידורי של הרשומה
55-60 תאריך העיבוד

את תאריך העיבוד מקבלים מכרטיס אשר בו מנוקב התאריך בטורים 1-6.
בסיום העבודה יש להדפיס במכונת הכתיבה של המפעיל את מספר רשומות הפלט (המספר הסידורי האחרון שהתקבל).

<u>תרשים מלבני (כללי):</u>



התוכנית:

```
    A   B

101 IDENTIFICATIØN DIVISIØN.
102 PRØGRAM-ID. 'INØUT'.

103 ENVIRØNMENT DIVISIØN.
104 INPUT-ØUTPUT SECTIØN.
105 FILE-CØNTRØL.
106     SELECT INP     ASSIGN  TØ  SYS021.
107     SELECT ØUTP    ASSIGN  TØ  SYS022.
108     SELECT KARTIS  ASSIGN  TØ  SYSIPT.

109 DATA DIVISIØN.
110 FILE SECTIØN.
111 FD  INP LABEL RECØRD IS STANDARD RECØRDING MØDE IS F
            BLØCK CØNTAINS 10 RECØRDS
113         RECØRD CØNTAINS 50 CHARACTERS
114         DATA RECØRD IS IN-AREA.
115 01      IN-AREA PIC X(50).

116 FD  ØUTP LABEL RECØRD IS STANDARD
117          RECØRDING MØDE IS F
118          RECØRD CØNTAINS 60 CHARACTERS
119          BLØCK CØNTAINS 20 RECØRDS
120          DATA RECØRD IS ØUT-AR.
201 01  ØUT-AR.
202     03 ØLD      PIC X(50).
203     03 SIDURI   PIC 9(4).
204     03 TAARICH  PIC 9(6).

205 FD  KARTIS LABEL RECØRD IS ØMITTED
206            RECØRDING MØDE IS F
207            RECØRD CØNTAINS 80 CHARACTERS
208            DATA RECØRD IS KK.
```

```
01  KK.
     02 DATE     PIC X(6).
     02 FILLER   PIC X(74).

WØRKING-STØRAGE SECTIØN.
77  MØNE PIC 9(4) VALUE ZERØ.

PRØCEDURE DIVISIØN.
       ØPEN INPUT INP KARTIS ØUTPUT ØUTP.
       READ KARTIS AT END GØ TØ LØØP.
LØØP.  READ INP AT END GØ TØ SØF.
           MØVE IN-AREA TØ ØUT-AR.
           ADD 1 TØ MØNE. MØVE MØNE TØ SIDURI.
           MØVE DATE TØ TAARICH.
       WRITE ØUT-AR. GØ TØ LØØP.
SØF.   CLØSE INP ØUTP KARTIS.
           DISPLAY 'TØTAL ØUTPUT RECØRDS=' MØNE.
       STØP RUN.
```

הסבר לתוכנית

א. מבנה התוכנית כולל את החטיבות לפי הסדר הקבוע שלהן בתוכנית.

ב. בחטיבה הראשונה - IDENTIFICATION - נמסר רק שם התוכנית - 'INOUT' בשורה 102. הגדרת שם התוכנית הינה חובה.

ג. בחטיבה השניה - ENVIRONMENT - נמצאת הגדרת הקבצים בפרק INPUT-OUTPUT תחת הכותרת FILE-CONTROL (בשורה 105). בתוכנית מוגדרים שלשה קבצים בשלש יחידות קלט/פלט ע"י הוראות SELECT לכל אחד מהם.
קובץ INP הוא קובץ הקלט ביחידה שכנוייה המוכר למערכת ההפעלה הוא SYS021 (הגדרה זו מתאימה למערכת הפעלה ספציפית במחשב י.ב.מ.) והיא מוגדרת ככונן סרט מגנטי מסויים. קובץ הפלט OUTP כינויו SYS022 והוא מוגדר גם כן על כונן סרט מגנטי, אחר. קובץ הכרטיסים KARTIS נקרא מיחידה שכינויה SYSIPT והיא מוגדרת כקוראת הכרטיסים.

ד. החטיבה השלישית - DATA - כוללת את תאור הקבצים והרשומות שלהם בפרק FILE (SECTION), ומונים ושטחי עבודה אחרים - בפרק WORKING-STORAGE.
לכל קובץ יש משפט FD המגדיר אותו, ולאחריו תאור מבנה הרשומה:

1) קובץ INP (שורה 111) הוגדר עם תויות סטנדרטיות, אורך רשומה קבוע (F), 10 רשומות לבלוק, 50 תוים לרשומה. שם הרשומה של הקובץ היא IN-AREA ותאורה ניתן בהמשך, בשורה 115. רשומה זו מכילה שדה אחד בן 50 תוים. בבעיה שלפנינו אין צורך לפרט יותר.

2) קובץ OUTP (שורה 116) הוגדר עם תויות סטנדרטיות, מבנה רשומה קבוע, רשומות באורך 60 תוים, 20 רשומות לבלוק. שם הרשומה של הקובץ הוא OUT-AR ותאורה ניתן בהמשך.
רשומת הפלט בשורה 201 OUT-AR כוללת פרוט של השדות המרכיבים אותה ולכן שמה הכללי ניתן ברמה 01 כנתון קבוצתי. פרוט השדות ניתן ברמה 03:
השדה הראשון OLD יכיל את תוכן רשומת הקלט אשר תועבר אליו, השדה SIDURI - את המספר הסידורי שיחושב, והשדה TAARICH - את תאריך העיבוד שיועבר מכרטיס מנוקב.

3) קובץ KARTIS (שורה 205) אינו מכיל תויות כותרת, רשומותיו באורך קבוע, באורך 80 תוים. שם הרשומה הוא KK. שים לב שלא צויין גודל הבלוק מכיון שהרשומות אינן מקובצות בבלוקים. הרשומה (שורה 209) KK מוגדרת כנתון קבוצתי ברמה 01. הפרוט כולל שני שדות: DATE (שאותו נעביר לרשומת הפלט) ושדה אחר שמוגדר כ-FILLER כי אינו חשוב לנו.

4) בפרק שטח העבודה (שורה 212) מוגדר שדה אחד המיועד להיות מונה למספר הסדורי של הרשומות.

5) שים לב שסדר הפסוקים במשפט FD אינו קבוע. אפשר להפריד פסוק ממשנהו בפסיק, אולם בסוף המשפט חייבת לבוא נקודה. תאור רשומת הקובץ חייב להופיע בהמשך למשפט FD שאליו היא שייכת.

ה. החטיבה הרביעית - PROCEDURE - כוללת את הוראות התוכנית.

1) ההוראה הראשונה (שורה 215) היא ההוראה OPEN לפתיחת הקבצים. בהוראה אחת אפשר לציין קבצי קלט ופלט, אולם יש לקבצם בסמוך למלת המפתח INPUT או OUTPUT בהתאמה.

2) הוראת הקריאה לכרטיס (216) תגרום להעברת תוכנו לרשומה KK שהוגדרה (שורה 209). אם מגיעים לסוף הקובץ מסתעפים לקטע LOOP. הכוונה זו אינה דרושה וניתנה לשם הסבר בלבד: מכיון שבמקרה שלפנינו יש צורך לקרוא כרטיס אחד בלבד, לא נחזור להוראת הקריאה לכרטיס ולכן לא נגיע למצב בו תתקבל התשובה של סיום הקובץ, וגם לא יתבצע הסיעוף לפי...AT END.

3) בשורה 217 מבוצעת קריאת רשומה מקובץ הקלט. כאן ההוראה ...AT END הינה רלוונטית. בסיום קובץ הקלט נעבור לקטע SOF שבו תבוצענה הפעולות של סיום התוכנית.

4) מהלך התוכנית מתואר בהוראות הבאות (218 עד 220): העברת רשומת הקלט לשטח הפלט, הוספת 1 למונה המספר הסידורי, העברתו לשדה SIDURI והעברת התאריך מרשומת הכרטיס לשדה הפלט המתאים. עכשיו אפשר לבצע את הוראת WRITE לקובץ הפלט (שורה 221) ולחזור ל-LOOP כדי להמשיך בקריאת רשומות הקלט. בהוראת הכתיבה מציינים את שם הרשומה בלבד, ללא שם קובץ.

5) כאשר הסתיימה קריאת כל רשומות הקלט עוברים לקטע SOF (שורה 222). מבצעים סגירת הקבצים. בהוראה CLOSE לא צויין סוג הקובץ אלא רק שמו מדפיסים במכונת הכתיבה לידיעת המפעיל את המספר הסידורי האחרון שהתקבל עם הסבר מתאים, ומסיימים בהוראת STOP RUN.

חלק חמישי:

תכנות פ.ל.1

PL/I

פרק ו: יסודות השפה-PL/I

הקדמה

שפת התכנות PL/I פותחה באמצע שנות הששים ע"י חברת י.ב.מ. וע"י איגוד המשתמשים בציוד י.ב.מ. בארה"ב. בפיתוח השפה הושם הדגש על שתי נקודות:

א. התאמה לעיבודים מינהליים ועיבודים מדעיים כאחת.

ב. ניצול מירבי של תכונות המחשבים החדשים מסדרת י.ב.מ./360 ו-י.ב.מ./370, בני הדור החדש.

השאיפה לכלול בשפה אחת את האפשרות לעיבודים מינהליים ומדעיים, כדוגמת האפשרויות הקיימות בשפות קובול ופורטרן, והוספת תכונות נוספות יצרו שפה רבת-מרכיבים. רבוי המרכיבים לא פגע בפשטות המבנה והלוגיקה של השפה, אך הוא דורש מהמשתמש לנהוג בתבונה בבחירת המרכיבים השונים הדרושים לו בכל מקרה.

ל-PL/I יש ניבים אחדים בהתאם למערכת ההפעלה ולמהדורה של המהדיר. החומר שלפנינו מבוסס על מרכיבי השפה הבסיסיים של המהדיר PL/I optimizer. מהדיר זה כולל את מירב התכונות החדשות של השפה והוא הנפוץ כעת. חשיבותו רבה ביכולתו לבצע אופטימיזציה של הקידוד המופק בתהליך ההידור כדי לקבל תוכנית יעילה ככל האפשר מבחינת תפושת הזכרון וזמני הביצוע.

שפת התכנות PL/I הינה שפת תכנות עילית, המאפשרת למתכנת ל"התבטא" במשפטי השפה האנגלית. אך גם כאן ישנן אפשרויות להרחיב ולקצר, וכאן אנו מגיעים לאחת התכונות של PL/I - הכתיבה התמציתית. במסגרת זו כלולים קיצורים של מלות מפתח של השפה, שימוש בסמלים וסימנים בהוראות ביצוע, והעיקר - שימוש ב"ברירת-מחדל". ברירת-המחדל (default option) הינה ברירה אשר מותר להשמיט אותה מהגדרה או הוראה מכיון שהמהדיר מציג תמיד אפשרות מוסכמת ידועה. למשל, אם לא מגדירים מבנה של נתון ההנחה היא שהוא במבנה דצימלי, אם לא מגדירים שיטת ארגון בקובץ מניחים שהוא במבנה סיקוונציאלי.

בחרנו להציג את שפת התכנות בדרך אשר תאפשר למתכנת להתחיל לכתוב תוכניות פשוטות, ע"י לימוד המרכיבים הבסיסיים ההכרחיים, ורק אח"כ הוספנו הרחבות המאפ-שרות פתרון בעיות מורכבות יותר. נסינו לשמור על העקרון של כתיבה בטכניקה

פשוטה אשר מבטיחה את מידת הבהירות של התוכנית ובעקיפין - גם את אמינותה ויעילותה.

מרכיבי השפה

ה ת כ נ ו ת

כתיבת תוכנית ב-PL/I הינה פשוטה ונוחה, בעיקר מכיון שהמגבלות על הקידוד הן מזעריות.

תוכנית PL/I בנויה מפסוקים (statements) המתבצעים לפי סדר כתיבתם בתוכנית. הפסוקים כוללים הוראות ביצוע, הגדרות לקבצים ונתונים והוראות פיקוח שונות (הוראות הסתעפות, הוראות התניה ועוד).

הפסוק מהווה צרוף של מלים (words), מפרידים (delimiters) בין המלים, מרווחים (blanks) והערות (comments), והוא חייב להסתיים בנקודה-פסיק (;).

דוגמאות לפסוקי תוכניות:

הגדרת נתון	DECLARE DATE CHARACTER (6);
פסוק השמה	MØNE = MØNE + A;
הוראת הסתעפות	GØ TØ START;
הוראת פלט	PUT LIST (NAME, ADDRESS);
הוראת התנייה	IF A = B ..... ;

את פסוקי התוכנית כותבים בדף הקידוד בצורה חופשית וללא מבנה מוגדר. דף הקידוד מקביל למבנה כרטיס מנוקב ומכיל 80 עמודות. את הפסוקים כותבים בעמודות 2 עד 72, ללא הגבלת מיקום; עמודה 1 אינה בשימוש ואסור לכתוב בה; בעמודות 73 עד 80 אפשר לכתוב קוד זהוי של התוכנית ומספר סידורי לשורות (כרטיסים). אפשר לגלוש בכתיבת פסוקים משורה לשורה, והכתיבה בשורה העוקבת יכולה להתחיל בכל עמודה שהיא.

פרטי התוכנית והתוכניתן		
1	2 .... פסוקי התוכנית 72	73 זהוי 80
//////	START: PROCEDURE OPTIONS (MAIN);	SAMPL 010
//////	GET DATA (A, B);	SAMPL 020
//////	A = B + 2;	SAMPL 030
//////	END;	SAMPL 040

תאור דף הקידוד מתיחס לכתיבת הוראות השפה בלבד. זכור כי נתונים לתוכנית אפשר לכתוב ולנקב בכל אחד מ-80 טורי הכרטיס.

קבוצת התוים של השפה

קבוצת התוים מכילה 60 תוים: תוים אלפבתיים בשפה האנגלית, ספרות עשרוניות, תוים מיוחדים.

א. 29 תוים אלפבתיים - הכוללים את תוי האלפבית האנגלי (אותיות "גדולות" בלבד) ושלשת התוים $, #, & לתכנות ב-PL/I משתמשים בתוים אלפבתיים בשפה האנגלית. קלט ופלט של תוכניות PL/I יכולים לכלול נתונים באלפבית עברי. האלפבית העברי מוצג במקום תוים באלפבית האנגלי לפי טבלה מוסכמת. להדפסת דוחו"ת בעברית יש לטפל גם בהפיכת כוון התוים כמוסבר בפרק "שפת התכנות קובול".

ב. 10 ספרות עשרוניות - 0 עד 9.

ג. 21 תוים מיוחדים - אשר לרבים מהם יש תפקידים בשפה, ועליהם נעמוד בהמשך, בהקשר לכל נושא.

מרווח (מסומן כ-ƀ בכתיבה)	(תו רווח)	סימן האחוז	%
סימן השויון או ההשמה	=	נקודה ופסיק	;
סימן החיבור	+	נקודותיים	:
סימן החיסור	-	סימן ה"לא"	¬
הכוכבית או סימן הכפל	*	סימן ה"ו-"	&
לוכסן או סימן החילוק	/	סימן ה"או"	|
סוגר שמאלי	(	סימן ה"גדול מ-"	>
סוגר ימני	)	סימן ה"קטן מ-"	<
פסיק	,	מקף (בגובה בסיס האות)	_
נקודה עשרונית או בינרית	.	סימן השאלה	?
גרש	'		

אפשרי צרוף של שני סימנים כדי ליצור משמעויות חדשות כמו. למשל <= מציין "קטן או שוה", או הצרוף של שתי כוכביות ** המציין "העלאה בחזקה".

מלים, מפרידים, מרווחים והערות

המלים של השפה שייכות לאחת משתי קטגוריות, מזהים (identifiers) וקבועים (constants). המזהים הם שמות של שדות ונתונים, תויות להוראות וכד' והקבועים הם ערכים מספריים.

התוים המיוחדים המוגדרים כמפרידים מחולקים לשתי קבוצות: למפרידים (separators) ואופרטורים (operators). מבין המפרידים נציין כאן במיוחד את התו נקודה-פסיק (;) המסיים פסוק PL/I, ואת הסוגרים התוחמים מלים בפסוקים מסוימים ומשרתים בבטויים במשמעותם הרגילה באלגברה. האופרטורים כוללים את התוים המשמשים להגדרת פעולות אריתמטיות ופעולות השוואה.

מלים עוקבות בפסוק אפשר להפריד ע"י מפריד אחד או יותר ו/או ע"י מרווחים. לדוגמה: FLDA יחשב כמזהה אחד אך הצרוף FLD A יחשב כשני מזהים בגלל המרווח המפריד בין שתי מלים; x+y יחשב כשני מזהים הקשורים ע"י סימן החיבור +, וזה זהה לצרוף y + x כאשר סימן השויון + מוקף במרווחים. את המרווח אפשר להוסיף בכל מקום בין מלים שונות או בין מלים ומפרידים. הערות אפשר לכתוב בכל מקום בו אפשר להשתמש במרווח, כלומר בין שתי מלים או בין מלים ומפרידים. להערה אין כל משמעות עבור המהדיר והיא מיועדת, כשמה, עבור המתכנת כהסבר בלבד. מבנה ההערה הוא: /* גוף ההערה */ דהיינו טכסט באורך תוים כלשהוא בין שני סימני ההפרדה המוכרים */ לציון תחילת הערה, ו- /* לציון סוף הערה.

לדוגמה: GØ TØ /* BRANCH TØ TAX CALCULATIØNS */ SUBA;

מ ז ה י ם.

מזהה (identifier) הוא מלה או קבוצת מלים המחוברות במקפים, והוא נוצר על ידי המתכנת כדי לתת שם או תוית לקובץ, לפסוק בתוכנית, לקבוצת פסוקים, לשדה ברשומה, ועוד.

יצירת המזהה נתונה בידי המתכנת וצריכה להעשות לפי כללי הכתיבה: מזהה עשוי להיות תו אלפביתי בודד או מחרוזת תוים אלפמריים (תוים אלפבתיים וספרות) ו/או מקף. משני צידיו של המזהה ישנם מרווחים או מפרידים אחרים מותרים. התו הראשון של המזהה חייב להיות אלפביתי. מספר התוים במזהה אסור שיהיה גדול מ-31.

דוגמאות למזהים:

A, FILE2, LØØP_3, SØF_HADPASAT_SHURA, #32

סוג אחר של מזהים כולל את מלות המפתח בשפת התכנות. מלת מפתח (Keyword) הינה מזהה, אשר בהופיעו בהקשר המתאים הינו בעל משמעות לתוכנית התרגום. מלת המפתח עשויה לציין הוראת פעולה או הגדרה של נתונים. דוגמאות:

GØ TØ, GET DATA, REAL, DECIMAL

לחלק ממלות המפתח יש גם שם מקוצר בו אפשר להשתמש במקום השם הרגיל. דוגמאות DCL במקום DECLARE, VAR במקום VARYING.

פסוקי התוכנית

פסוק בתוכנית PL/I משמש להגדרה, הוראת ביצוע, או להוראת בקרה. הפסוק יכול להיות פשוט או מורכב בהתאם למכלול הפעולות שהוא מבצע.

*פסוק פשוט מכיל את גוף הפסוק והוא מסתיים במפריד נקודה-פסיק (;). סוגי הפסוקים:

א. פסוק עם מלת מפתח (Keyword statement) מכיל מלת מפתח (אחת או יותר) לציון ההוראה או התפקיד של הפסוק. לדוגמה:

מלת המפתח: GØ TØ	GØ TØ START;
שתי מלות מפתח: FILE, ØPEN	ØPEN FILE (TALMIDIM);

ב. פסוק השמה (assignment statement) מכיל את סימן ההשמה (=) ואינו מכיל מלת מפתח. לדוגמה:

A = B + C;

MØNE = TØTAL;

ג. פסוק ריק (null statement) מכיל רק את המפריד נקודה-פסיק (;) ומשמעותו "אין פעולה".

לדוגמה פסוק השמה ופסוק ריק; A = B + C;;

* פסוק מורכב מכיל פסוקים פשוטים אחדים המהווים את "גוף הפסוק" ולאחריהם יש נקודה-פסיק. בקבוצה זו כלולים פסוקי ההתנייה IF ו-ON. לדוגמה:

IF A > B THEN A = B + C;

פסוק ההשמה A = B + C מוכל בתוך הפסוק המורכב IF.

* קבוצות (groups). קיימים מצבים בהם נדרש לראות קבוצת פסוקים כיחידת ביצוע אחת. לשם כך הם מוגדרים כקבוצה כאשר בראשם הפסוק DO ובסיומם הפסוק END. לדוגמה:

```
DØ;
      (פסוק 1);
      (פסוק 2);
END;
```

לכל אחד מסוגי הפסוקים אפשר להוסיף קידומת (statement prefix). הקידומות משמשות כתויות זהוי (label prefix) או כקידומות התנייה (condition statements).

תפקיד התוית לזהות את הפסוק כדי שאפשר יהיה להתיחס אליו במקום אחר בתוכנית, כמו למשל לצורך סיעוף. התוית נרשמת לפי כללי כתיבת "מזהה", לפני המשפט והיא מופרדת ממנו בנקודותיים (:). לדוגמה,

```
K1 : Z = FLD_A + FLD_B;
```

אפשר לרשום לפסוק אחד יותר מתוית אחת, לדוגמה:

```
LØØP: TEST_DATE: IF A > B ... ;
```

בנושא קידומות התנייה (condition prefixes) נדון בשלב מאוחר יותר.

<u>מבנה תוכנית</u>

פסוקים של השפה מיועדים לקריאת או כתיבת נתונים, לביצוע פעולות אריתמטיות, לבצע השואות ולפעול בהתאם למסקנות ההשואה והם כוללים גם הוראות הגדרה לקבצים ונתונים. פסוקים אחרים מיועדים לפיקוח על מסלול ביצוע התוכנית.

התוכנית מורכבת, מאוסף של פסוקים המוגדר כבלוק (block). תוכנית אחת יכולה להבנות מכמה אוספי פסוקים כאלה, או מכמה בלוקים, מסוגים שונים. בכל תוכנית קיים בלוק <u>אחד</u> המוגדר בצורה הבאה:

```
(תוית 1): PRØCEDURE ØPTIØNS (MAIN);
          ... פסוקי התוכנית ....
          ... פסוקי התוכנית ....
          END (תוית 1),
```

בראש הבלוק מופיע הפסוק PROCEDURES, אשר במקרה זה מלווה בהגדרה OPTIONS (MAIN) לציון העובדה שזהו הבלוק הראשון בתוכנית, או <u>תחילת התוכנית.</u> לפסוק זה רשומה תוית זהוי שהיא שם הבלוק או שם התוכנית. בסיום הבלוק מופיע המשפט שבו מלת המפתח END ואשר יכול לכלול את תוית הזהוי של הבלוק בהמשך גוף המשפט, אולם אין זה חובה.

דוגמת תוכנית PL/I

מטרת התוכנית לקרוא מספר שדות מכרטיס מנוקב. אם הערך של השדה N הוא אפס, יש להדפיסו, ואם איננו שוה לאפס יש לחשב את סכום השדות A, B, C, D ולחלקו ב-N ולהדפיס את התוצאה.

שורה

```
1  SAMPLE: PRØCEDURE ØPTIØNS (MAIN);
2          GET DATA (A,B,C,D,N);
3          IF N = 0 THEN PUT DATA (N);
4                    ELSE DØ;
5                         SN = (A+B+C+D)/N;
6                         PUT DATA (SN);
7                         END;
8          STØP;
9  END SAMPLE;
```

בשלב זה נכתבו רק הוראות הביצוע המטפלות בפתרון הבעיה. שים לב לצורת הכתיבה: המבנה שבו נכתב קטע התוכנית נקבע על ידי המתכנת לנוחיותו. בדרך כלל מעדיפים לכתוב רק פסוק אחד בשורה ולכתוב פסוקים שונים בהזזה לשם בהירות.

ה ס ב ר

- בשורה 1 רשום פסוק הפתיחה של הבלוק הראשון של התוכנית. יש לו תוית SAMPLE, שזהו גם שם התוכנית.
- בשורה 2 רשומה הוראת קריאה של 5 שדות מתוך הכרטיס. בפסוק זה ישנה מלת המפתח GET DATA.
- בשורה 3 ישנו פסוק מורכב (פסוק התנייה) שבו נבדק ערכו של N. אם N שוה ל-0 מדפיסים אותו לפי הוראת הפסוק PUT DATA. אם N איננו שוה ל-0 מדלגים על הוראת ההדפסה בשורה 3 ועוברים לביצוע הפסוק שבשורה 4.
- בשורה 4 מופיע הפסוק DO, המציין תחילתה של קבוצת פסוקים (בשורות 5, 6) שיש לבצע אותה כהמשך להוראת ההתנייה ELSE (כאשר $N \neq 0$). בסוף קבוצה זו מופיע המשפט END (שורה 7).
- בשורות 6,5 מופיעה הוראת החישוב של הערך SN והדפסתו.
- בשורה 8 ישנה ההוראה STOP המפסיקה את פעולת התוכנית (אפשר להשמיטה אם היא האחרונה בבלוק).
- בשורה 9 רשום הפסוק לציון סיום הבלוק/התוכנית.

פרק 2: הגדרת נתונים

סוגי נתונים

אפשר לסווג את ההוראות המבוצעות ע"י המחשב לשלוש קבוצות:

א. הוראות לוגיות - המטפלות ברצף של תוים (characters) או ברצף של סביות (bits),

ב. הוראות אריתמטיות - המטפלות בנתונים מספריים,

ג. הוראות פיקוח - הקובעות את סדר ביצוע ההוראות בתוכנית.

לשלוש קטגוריות אלו של סוגי הוראות, אפשר לסווג בהתאמה את סוגי הנתונים: נתונים במחרוזות (strings), נתונים מספריים (numeric data) ונתוני תויות (labels). לכל סוג של נתונים ישנם מאפיינים המגדירים את תכונותיו.

נתונים במחרוזות

המחרוזות כוללות שתי קבוצות נתונים: נתוני תוים (CHARACTER) ונתוני ביות (BIT).

מחרוזות התוים כוללות בעיקר את נתוני הקלט המוזנים מאמצעים חיצוניים (לדוגמה כרטיס מנוקב) ואת נתוני הפלט המועברים להדפסה. נתונים שונים בקבצי המחשב אשר אינם משמשים לפעולות חשבוניות (למשל שם, כתובת, תאריך, קודים) נשמרים בדרך כלל בצורת מחרוזות תוים. אך נתונים מספריים אשר מיועדים לפעולות חישוב חייבים לעבור הסבה אל צורת הייצוג הדרושה לפעולות חשבוניות (דצימלי, או בינרי). נתונים מספריים אחרים יכולים גם כן לעבור הסבה לפי הענין.

מחרוזות הסביות מייצגות תנאי כן, לא (0/1) או OFF/ON, והן משמשות בעיקר בפעולות לוגיות.

דוגמאות:

המחרוזת כוללת צרוף כלשהוא של תוים או סביות, בהתאם לענין, המוצגים בין שני גרשים. תכונת אורך המחרוזת נקבעת על פי מספר התוים או הסביות אשר בין הגרשים.

במחרוזת תוים מותר כל צרוף של תוים של השפה:

'ABC' , 'TEL-AVIV', '45.8' , 'XYZ *B-3'

אם אחד מתוי המחרוזת הוא <u>גרש</u> יש לרשום אותו פעמיים, בצורה הבאה: A'B ירשם במחרוזת כ- 'A''B'

במחרוזת סביות מותר כל צרוף של הספרות 0, 1 ובנוסף יש לרשום את האות B מימין לגרש הימני התוחם את המחרוזת:

'1110001'B , '100110'B

כאשר ישנה במחרוזת סדרת תוים או סביות החוזרת על עצמה אפשר לרשום כך:

'ABABAB' ⟶ (3)'AB'

כלומר, מחרוזת תוי AB מופיעה 3 פעמים. גורם הכפל מופיע בסוגרים <u>לפני</u> המחרוזת.

לדוגמה: '101'B (2) ⟶ '101101'B.

<u>נתונים מספריים</u>

הנתונים המספריים מייצגים ערכים המשמשים בפעולות אריתמטיות, והם כלולים בשתי קבוצות עיקריות: נתונים בינריים ונתונים עשרוניים. כל קבוצה כזו מחולקת לקבוצות משנה: הצגה בנקודה קבועה והצגה בנקודה צפה.

הנתונים העשרוניים מוצגים ע"י רצף של ספרות עשרוניות, כל אחת בת 4 סביות. הנתונים הבינריים מציגים את ערכי המספרים בשיטה הבינרית ע"י רצף של סביות (0,1).

מבנה המספר יכול להיות עם <u>נקודה קבועה</u> דמיונית. בשיטה זו רואים את המספר כמי שמכיל ערך שלם בלבד או מספר קבוע של ספרות מימין לנקודה. כל פעולה שתעשה עם המספר תתיחס להגדרת המקום של הנקודה, אשר <u>אינה</u> מופיעה בזכרון המחשב, אולם המהדיר "זוכר" היכן היא נמצאת (ולכן היא "דמיונית"). שיטת החישוב עם נקודה קבועה מקבילה לשיטת החישוב הידנית.

המספרים המוגדרים בשיטת <u>הנקודה הצפה</u> בנויים משני חלקים: מנטיסה ואקס-פוננט. שיטת ייצוג זו נהוגה בפעולות אריתמטיות בעבודות מדעיות בלבד.

<u>ד ו ג מ א ו ת:</u>

מספרים עשרוניים יופיעו בצורת כתיבתם, ללא תוספת, אך מספרים בינריים ירשמו בצורת כתיבתם בתוספת האות B מימין למספר. דיוק המספר נקבע על פי מספר הספרות שלו ומיקומה של הנקודה העשרונית.

מספרים עשרוניים עם נקודה קבועה:	25.3	-0.45	1975
מספרים בינריים עם נקודה קבועה:	11100B	1.1101B	101B

מספרים בשיטת הנקודה הצפה יוגדרו כך:

מספרים עשרוניים:	12E3	שערכו	12000
	4.5E-2	שערכו	0.045
מספרים בינריים:	1.101E-2B	שערכו	0.01101B
	111E3B	שערכו	111000B

<u>נתוני תויות</u>

התויות מיועדות בדרך כלל לתת שם לפסוקים שיש לפנות אליהם במהלך התוכנית ע"י הוראות סיעוף. עם סיום הידור התוכנית מציינת התוית בדרך כלל כתובת ממשית בזכרון שבה נמצאת הוראת המחשב. כאשר תויה מוגדרת כנתון - אפשר לשנות את ערכה במהלך התוכנית, ולפיכך אם ישנה הוראת סיעוף אל תוית כזו, יבוצע מדי פעם סיעוף אל כתובת אחרת בהתאם לערכה המשתנה של התוית באותו זמן.

תויות משמשות גם כשמות לנתונים (names of data items) המופיעים בתוכנית. אלו יכולים להיות נתונים בשדות בודדים או נתונים המאוגדים בשדות במבנה רשומה. במקרה זה אפשר לתת שמות לרשומה כולה או לרלקים שלה בהתאם לצורך.

<u>דוגמאות</u>

נתוני תויות מוגדרים כמזהים.

SØF היא נתון-תוית. SØF: STØP;

ID, NAME, ADDRESS, AGE, STATUS - הם נתוני-שדות ברשומה של פרטים אישיים.

קבועים ומשתנים

נתונים בתוכנית יכולים להופיע באחת משתי צורות כמשתנה (Variable) או כקבוע (Constant).

<u>המשתנה</u> הוא שם סמלי לכתובת זכרון אשר מכילה את הנתון ואשר ערכו, ובהתאמה - ערכו של המשתנה, עשוי להשתנות במהלך ביצוע התוכנית. בחירת השם הסמלי הינה ברשות המתכנת וכפופה לכללי כתיבת מזהים.

<u>קבוע</u> הוא נתון המופיע בתוכנית בערכו הממשי ואינו ניתן לשינוי במהלך הביצוע. <u>תכונות הקבוע</u> משתמעות מצורת כתיבתו בתוכנית, כקבוע של מחרוזת תוים,

קבוע מספרי או כקבוע תוית.

הפסוק הבא מתאר ביטוי אריתמטי והוא מכיל משתנים וקבועים:

```
X1 = PRT + 5.5 * K - 10.5;
```

X1, PRT , K הם משתנים; 5.5, 10.5 הם קבועים. משמעות הפסוק כמשמעותו באלגברה, כאשר התוצאה הסופית מתקבלת במשתנה X1.

בניגוד לקבועים המוגדרים בתוכנית, יש צורך להגדיר במפורש את התכונות של המשתנים, וזאת נעשה ע"י הפסוק DECLARE. מטרת ההגדרה להציג את התכונות בזמן הידור התוכנית כדי שהמהדיר יוכל להשתמש בהוראות המתאימות בהתאם לסוג ואופי הנתונים.

פסוק ההכרזה - DECLARE - לנתונים

הכרזה של תכונות נתונים נעשית ע"י פסוק ההכרזה DECLARE (או בקיצור DCL). אפשר להגדיר תכונות נתונים בקלט ופלט גם בצורה שונה, עליה נעמוד בהמשך. הגדרת התכונות של הנתונים נעשית ע"י מלות מפתח של השפה. מבנה פסוק ההכרזה:

DECLARE (שם 1) (תכונות 1) , (שם 2) (תכונות 2) , ... ;

"שם 1", "שם 2" הם שמות משתנים בתוכנית.
"תכונות" הם רשימת מלות המפתח (אחת או יותר) המגדירות את תכונות המשתנה. בין מלות המפתח יש להפריד במרווח אחד לפחות.
בין הגדרת תכונות של משתנים יש להפריד בפסיק. בסיום הפסוק יש לרשום נקודה-פסיק.
<u>הערה</u>: את פסוק ההכרזה אפשר לרשום <u>בכל מקום בתוכנית</u>, גם בין פסוקי הוראות.
דוגמה:

```
DECLARE PARIT FIXED DECIMAL (8),
        DESCRIPTIØN CHARACTER (30),
        PRICE FIXED DECIMAL (6,2);
```

בדוגמה זו PARIT, PRICE, DESCRIPTION הם שמות משתנים המוגדרים בפסוק ההכרזה. התכונות הן: סוג הנתון - FIXED DECIMAL, CHARACTER ובסוגרים - מספר התוים/ספרות של המשתנה.

<u>תכונות משותפות</u>

פסוק ההכרזה מאפשר להכריז על קבוצת משתנים בעלי תכונות זהות ללא צורך בחזרה על תכונותיו של כל משתנה. מבנה פסוק ההכרזה הוא:

DECLARE (שם 1, שם 2, שם 3 ...) (תכונה א),
(שם) (תכונה ב) ... ;

לדוגמה:

DECLARE (SHEM, KTØVET) CHARACTER (25);

הכרזת תכונות משותפות יכולה להעשות בצורות נוספות אשר לא נעמוד כאן על כולן.

<u>ערך התחלתי</u>

התכונה INITIAL (או בקיצור INIT) אשר מוזכרת בפסוק ההכרזה מציינת ערך התחלתי (קבוע) אשר ינתן למשתנה.

מבנה ההוראה DECLARE (משתנה) (תכונה) INITIAL (קבוע);

קבוע אשר מופיע בתכונת INITIAL יכתב לפי כללי כתיבת קבועים (ראה בהמשך). לדוגמה:

DECLARE SUM FIXED DECIMAL (3) INIT (0),
NAME CHARACTER (4) INIT ('JOHN');

כלומר, הערך של משתנה SUM יהיה אפס, והערך של NAME יהיה JOHN.

בשלב זה נעמוד בעיקר על הגדרת התכונות של שני סוגי נתונים: מחרוזות תוים ומספרים עשרוניים עם נקודה קבועה.

הגדרת תכונות מחרוזות

מחרוזות אפשר להגדיר כתוים - <u>CHARACTER</u> (או בקיצור CHAR) או כסביות -<u>BIT</u>. תו במחשבי י.ב.מ. הינו צרוף של 8 סביות המוצגים ב-byte (בית) אחד בזכרון המחשב, והוא יכול, כנתון, להכיל צרופים נוספים לאלו המותרים והמוכרים בקבוצת התוים של השפה. נתונים המוגדרים כתוים הינם במבנה מתאים להדפסה ואינם דורשים כל עריכה שהיא.

הגדרת המחרוזת נעשית בפסוק ההכרזה DECLARE בצורה הבאה:

$$\text{DECLARE} \quad (\text{שם 1, שם 2, ...}) \begin{Bmatrix} \text{CHAR} \\ \text{BIT} \end{Bmatrix} (n);$$

כאשר "שם 1", "שם 2" וכו' הם שמות משתנים, ו-n הוא מספר עשרוני המציין את מספר הסימנים במחרוזת אחת או בכל אחת מהמחרוזות הנמצאות ברשימת המשתנים. לכל המחרוזות ברשימה ישנן אותן התכונות.

לדוגמה פסוקי הכרזה עם משתנה מחרוזת אחד:

DECLARE NAME CHARACTER (25);

- שדה NAME מכיל 25 תוים.

```
DECLARE CØDES BIT (18);
```

- שדה מחרוזות בינרי CODES מכיל 18 סביות.

לדוגמה פסוקי הכרזה לקבוצת משתנים בעלי תכונה משותפת. בפסוק הכרזה אחד אפשר לכלול שתי הגדרות שונות:

```
DECLARE (NAME, ADDRESS) CHAR (30), (CØDE, MASK) BIT (4);
```

- המשתנים NAME ו-ADDRESS הם מחרוזות תוים באורך 30 כ"א,
- המשתנים CODE, MASK הם מחרוזות סביות באורך 4 כ"א. שים לב שלכל המשתנים באותה רשימה יש תכונות זהות.

אם אורך המחרוזת אינו קבוע, מציינים בפסוק ההכרזה את האורך המקסימלי האפשרי ואת מלת המפתח VARYING. לדוגמה, שדה ADDRESS אשר יכול להכיל עד מקסימום 50 תוים יוגדר בצורה הבאה:

```
DECLARE ADDRESS CHAR (50) VARYING;
```

הגדרת ערך התחלתי למשתנה יכול להעשות ע"י התכונה INITIAL, לדוגמה:

```
DECLARE STREET CHAR (6) INIT ('MODEIN'),
        MASK BIT (5) INIT ('10110'B);
```

אם הקבוע גדול באורכו מאורך המשתנה - יתווספו תוי רווח מימין, ואם הוא ארוך יותר - יקוצצו תוים מימין. אפשר להשתמש בגורם הכפלה כאשר תו או מספר תוים חוזרים במחרוזת, לדוגמה:

```
DCL DUGMA CHAR (100) ((50)'AB');
```

כלומר התוים AB יחזרו 50 פעם במשתנה DUGMA באורך 100 תוים.

תכונה נוספת למחרוזות נקבעת על ידי מלת המפתח PICTURE, אשר מיועדת בעיקר להגדרת נתונים מספריים הערוכים להדפסה, או לקלט נתונים. את ההסבר של תכונה זו נדחה לפרק הדן בעריכת נתונים.

תכונות של נתונים מספריים

לנתונים מספריים המשמשים בפעולות חשבוניות ישנם המאפיינים הבאים:

DECIMAL - מספר המוצג בשיטה עשרונית (בקיצור - DEC).

או BINARY - מספר המוצג בשיטה בינרית (בקיצור - BIN).

FIXED - מספר בשיטת הנקודה הקבועה

או FLOAT - מספר בשיטת הנקודה הצפה

וגם: precision (דיוק) - מספר הספרות של המספר. בשיטת הנקודה הקבועה מציינים

גם את מספר הספרות שמימין לנקודה.

לדוגמה: מספר עשרוני TOTAL שערכו 38795.46 המכיל 7 ספרות מהן 2 ספרות עשרוניות יוגדר כך:

DECLARE TØTAL DECIMAL FIXED (7,2);

כאשר המספר איננו מכיל ספרות עשרוניות אפשר להשמיט את המספר המציין את הספרות העשרוניות. הגדרת שדה בינרי: MØNE BINARY FIXED (20 ,2)

ברירת המחדל

ב-PL/I אין צורך להגדיר תמיד את תכונותיו של כל משתנה בתוכנית. תכונות של משתנים שונים יכולות להיות מובנות מתוך ההקשר או מצורת כתיבתם בתוכנית. במקרים מסוימים אפשר להגדיר רק חלק מהתכונות, מכיון שמלכתחילה נקבעה להם תכונה משלימה אם לא מגדירים תכונה אחרת. כאשר לא מגדירים את כל תכונות המשתנים, מניחים שתכונות מסוימות מיוחסות להם. זוהי "ברירת המחדל" או "default option".

א. כאשר אין הגדרת תכונות למשתנה, נקבעות תכונותיו לפי האות הראשונה של השם:
- אם האות הראשונה היא אחת מהאותיות I עד N המשתנה יוגדר כ-BINARY FIXED (כמו בפורטרן). למשל NUMBER, IR, M345.
- בכל שאר המקרים הגדרת המשתנה היא DECIMAL FLOAT.

. אם מציינים רק DECIMAL או BINARY, ההנחה היא שהמספר מסוג FLOAT.

ג. אם מציינים רק FIXED או FLOAT, ההנחה היא שהמספר מסוג DECIMAL.

קיימת ברירת מחדל גם עבור דיוק המספרים, השימוש בברירה זו איננו מומלץ למתחילים.

בהמשך נדון בנתונים עשרוניים בשיטת הנקודה הקבועה. שיטת ייצוג זו עונה על השמושים ביישומים ניהוליים, והיא גם נוחה להצגה. המשתמש בשפה ליישומים מדעיים יבחר בדרך כלל את השיטה הבינרית בנקודה קבועה או בנקודה צפה.

הגדרת נתונים עשרוניים עם נקודה קבועה

מבנה הנתונים

שיטת ההצגה העשרונית המוזכרת כאן היא Packed Decimal, שבה מקבצים שתי ספרות לתוך byte (בית) אחד בן 8 סביות. חצי הבית הימני ביותר של השדה מכיל את הסימן של המספר. מכיון ששדה יכול להכיל מספר שלם של בתים יש לחשב את גודלו

לפי מספר הספרות בצורה הבאה: מחלקים את מספר הספרות ב-2 ומוסיפים 1. אם מספר הספרות היה איזוגי ינוצל כל אורך השדה, אך אם היה זוגי - תתוסף הספרה אפס בחצי השמאלי של הבית השמאלי בשדה. בשדה אין נקודה עשרונית פיזית (כלומר - תו). ידיעת מקומה של הנקודה הינה באחריותו של המהדיר השומר נתון זה בטבלה שהוא מכין. לדוגמה:

המספר 2741 +

0 , 2	7 , 4	1 , +
בית 3	בית 2	בית 1

אורך השדה - 3 בתים.

ייצוג כל ספרה או סימן הוא ב-4 סביות.

<u>מבנה פסוק ההכרזה</u>

הצורה הכללית של פסוק ההכרזה עבור משתנים עשרוניים עם נקודה קבועה:

DECLARE (שם-משתנה) FIXED DECIMAL (P,q),
(שם-משתנה) DECIMAL (P,q) FIXED, ... ;

בפסוק הצהרה אחד אפשר להצהיר על יותר ממשתנה אחד. לאחר "שם-המשתנה" הנקבע ע"י המתכנת, תבוא רשימת תכונות, בסדר כלשהו. רשימה זו המכילה את מלות המפתח FIXED ו-DECIMAL וכן זוג מספרים עשרוניים בתוך סוגריים. זוג המספרים קובעים את <u>דיוק</u> הנתון בזכרון. P מציין כמה ספרות יכיל הנתון, ו-q מציין כמה מהספרות הן מימין לנקודה העשרונית. הנקודה העשרונית אינה חלק מן הנתון עצמו, דהיינו אינה תופסת מקום פיזי בזכרון - היא "נקודה מדומה", או "נקודה לוגית". "זכירת" מקום הנקודה העשרונית בכל אחד מן הנתונים נעשית ע"י המהדיר.

בין מלות המפתח בפסוק ההצהרה המגדירות את התכונות יהיה לפחות מרווח אחד. אם רוצים להצהיר על יותר ממשתנה אחד בפסוק ההצהרה אחד יש להפריד כל הצהרה ע"י פסיק. ערכו המקסימלי של P - מספר הספרות בשדה - המותר במחשב י.ב.מ./370 הוא 15. ערכו של q אינו מוגבל כפי שיוצג בהמשך.

להלן מספר דוגמאות לאפשרויות שונות של הכרזת הנתונים:

<u>דוגמה 1</u>:

DECLARE AVERAGE FIXED DECIMAL (7,2);

המשתנה AVERAGE מוגדר כעשרוני עם נקודה קבועה ודיוק (7,2). דהיינו, המספר המירבי של ספרות ב-AVERAGE הוא 7, מהן שתיים מימין לנקודה העשרונית.

כל נתון שיועבר למשתנה AVERAGE יוסב למבנה המתאים לנתון עשרוני עם נקודה קבועה. החלק השלם של הנתון יועבר ל-AVERAGE ספרה ספרה מימין לשמאל, החל מן המקום המדומה של הנקודה העשרונית. השבר העשרוני יועבר ב-AVERAGE ספרה ספרה משמאל לימין, החל מן המקום המדומה של הנקודה העשרונית.

אם מספר ספרות השלם קטן מזה המיועד, יושלם המקום הנותר באפסים מובילים (משמאל). אם מספר ספרות השבר קטן מן המקום המיועד, יושלם הנתון באפסים נגררים (מימין). למשל, אם נעביר ל-AVERAGE את המספר 1234.5, יכיל השדה את הנתון בצורה הבאה: +01234^50, כאשר החץ מורה על המקום הלוגי של הנקודה העשרונית. שים לב שבשדה סה"כ 8 תוים: 7 ספרות+סימן, וארכו - 4 תוים.

אם מספר הספרות של החלק השלם ו/או של השבר העשרוני גדול מן המוגדר, יקוצצו ספרות משמאל (מן החלק השלם) או מימין (מן השבר). למשל: אם נעביר ל-AVERAGE את הנתון -123456.789, יהיה ערכו של AVERAGE לאחר ביצוע ההשמה -23456^78. כתוצאה מאובדן ספרות של החלק השלם (ספרות משמעותיות) מתקבל <u>מצב-חריג</u> (exception). בזמן ביצוע התוכנית. אפשר להתנות עצירת התוכנית או סיעוף לשגרת בדיקה במצב כזה.

<u>דוגמה 2</u>:

DECLARE INTEREST DECIMAL (6,0) FIXED, RATE FIXED (6) DECIMAL;

למשתנים INTEREST ו-RATE דיוק זהה - 6 ספרות שלימים - למרות צורת הקידוד השונה. אפשר לותר על q אם הנתון שלם (כלומר q = 0).

<u>דוגמה 3</u>:

DECLARE CASH FIXED DECIMAL (4,-3);

ערך שלילי של q מציין שהמקום הלוגי של הנקודה העשרונית הוא שלשה מקומות <u>מימין</u> לספרה הימנית של הנתון. זהו מצב שבו שומרים רק את הספרות החשובות של המספר והספרות הימניות הן תמיד אפסים. למשל, המספר CASH הוא 1234 אולם ערכו האמיתי הוא 1234000. בנושא זה ראה הדיון במספרים בשיטת הנקודה הצפה (בחלק פורטרן).

<u>דוגמה 4</u>:

DECLARE DIVIDEND FIXED (2,5) DECIMAL;

בדוגמה זו, גדול ערכו של q מזה של P. הדבר מציין שמקום הנקודה העשרונית הוא q

מקומות משמאל לספרה הימנית של הנתון. כלומר, כאן נשמרות הספרות החשובות של המספר (שערכו קטן) והספרות משמאל הן אפסים. למשל, המספר בשדה DIVIDEND הוא 12 אולם ערכו האמיתי הוא 0.00012. ראה גם הדיון במספרים בשיטת הנקודה הצפה (בחלק פורטרן).

תכונות של קבועים מספריים

קבועים עשרוניים עם נקודה עשרונית קבועה מוצגים ב-PL/I בדרך המקובלת, לדוגמה: 9, 23.56-, 495.0. תכונותיהם של מספרים אלו נובעות מצורת הצגתם בתוכנית: DECIMAL FIXED. מידת הדיוק נקבעת לפי מספר הספרות, לדוגמה:

דיוק הקבוע 9 הוא (1),
דיוק הקבוע 23.56- הוא (4,2),
והדיוק של 495.0 הוא (4,1).

הכרזת ערך התחלתי

אחת מתכונות המשתנים הינה הערך ההתחלתי שלהם. ההכרזה נעשית ע"י התכונה INITIAL כבדוגמה הבאה:

```
DECLARE MAX FIXED DECIMAL (4,2) INITIAL (99.99),
        MØNE FIXED DECIMAL (6) INIT (0);
```

כלומר, MAX יקבל את הערך 99.99 והמשתנה MONE יקבל את הערך 000000 (6 אפסים). שים לב שהקבוע הרשום כתכונה של "הערך ההתחלתי" צריך להכתב לפי כללי רתיבה של קבועים מספריים.

פרק 3: מרכיבי תכנות בסיס״ם

ביטויים חשבוניים

ביטוי (expression) מכיל צרוף של קבועים, משתנים ואופרטורים. קיימים בטויים מסוגים שונים, ואחד מסוגים אלו הוא הביטוי החשבוני בו האופרטורים מייצגים את הפעולות החשבוניות.

הערכה של ביטויים (Evaluation of Expression)

חישובים נעשים ב-PL/I באמצעות השימוש בחמש פעולות החשבון הבסיסיות: חיבור (+), חיסור (-), כפל (*), חילוק (/) ופעולות מעריכיות - העלאה בחזקה (**). השימוש באופרטורים של כפל, חילוק ופעולה מעריכית, זהה לחלוטין לזה הנהוג בחשבון, אך לכל אחת מפעולות ה-(+) וה-(-) עשויים להיות שני מובנים: ציון סימן של איבר או קישור של שני איברים.

כאשר האופרטורים (+,-) משמשים לקישור של שני איברים בביטוי חשבוני, למשל A+B, נקראת פעולה זו infix operation. כאשר משתמשים באחד משני אופרטורים אלו משמאל לאיבר בודד של ביטוי, כגון X-, נקראת פעולה זו prefix operation. לדוגמה: ה-(+) בביטוי +Y-X**K הוא prefix וכזה הוא ה-(-) בביטוי הבא: 2*A**-3. בביטוי X+-Y יהיה ה-(+) infix וה-(-) prefix.

ניתוח והערכה של ביטוי חשבוני נעשה בתהליך הבא: הביטוי כולו, החל מהתו הראשון שמימין לסימן השויון וגמור במפריד (;) שבסוף הפסוק, נסרק משמאל לימין כדי לקבוע באיזה פעולות משתמשים ובאיזה סדר. הערכת הביטוי מבוצעת על פי סדר עדיפות קבוע. הפעולות בעדיפות ראשונה מתבצעות לפי הסדר מימין לשמאל, והפעולות בעדיפויות אחרות - משמאל לימין:

עדיפות ראשונה - העלאה בחזקה (**), prefix (+,-)
עדיפות שניה - כפל (*), חילוק (/)
עדיפות שלישית - infix חיבור (+) וחיסור (-)

תוצאות הביניים של התהליך נשמרות עד להשלמת הערכת כל הביטוי וקבלת התוצאה הסופית.

דוגמת פסוק עם ביטוי חשבוני: A = B + (-C) * 3

סדר העדיפות הוא: - (prefix), *(כפל), +(חיבור).

דוגמה

נתון הביטוי הבא. יש לחשב את ערכו.

-Y**4+K*X*B/D**2-4*C1/K*X;

1) פעולות בעדיפות ראשונה בביטוי הם Y-, 4**Y- ו-2**D. פעולות בעדיפות ראשונה יבוצעו מימין לשמאל, ומכאן, ראשית יבוצע D^2 ותתקבל התוצאה (החלקית) t1.
2) הפעולה הבאה בעדיפות ראשונה שתבוצע היא Y^4 והתוצאה - t2.
3) הפעולה האחרונה בעדיפות ראשונה היא הפיכת הסימן של t2.
4) כיון שלא נותרו פעולות בעדיפות ראשונה, יחזור הפיקוח לתחילת הביטוי כדי לבצע את הפעולות בעדיפות שניה משמאל לימין. ראשית תתבצע המכפלה K*X ותוצאת הביניים תשמר ב-t3.
5) t3 יוכפל ב-B והתוצאה תשמר ב-t4.
6) t4 יחולק ב-t1 (D^2) התוצאה תשמר ב-t5.
7) בשלב זה תתבצע המכפלה 4*C1 והתוצאה תשמר ב-t6.
8) t6 יחולק ב-K והתוצאה תשמר ב-t7.
9) t7 יוכפל ב-X והתוצאה תשמר ב-t8.
10) מאחר שאין יותר פעולות בעדיפות שניה יחזור הפיקוח לתחילת הביטוי לביצוע הפעולות בעדיפות שלישית, משמאל לימין. ראשית יחובר t2 ל-t5 והתוצאה תשמר ב-t9.
11) t8 יחוסר מ-t9 והתוצאה תשמר ב-t10.
12) הפעולה האחרונה היא השמת התוצאה t10 למשתנה המכיל את התוצאה הסופית של הערכת הביטוי.

השימוש בסוגריים

כאמור, הערכת הביטוי תתבצע לפי סדר כתיבתו ולפי כללי העדיפות לביצוע הפעולות השונות. לעתים יש צורך לשנות את סדר ביצוע הערכת הביטוי, ואז משתמשים בסוגריים. הסוגריים מחלקים את הביטוי השלם לביטויים חלקיים ומאפשרים הערכה של כל חלק בנפרד. התוצאה הסופית היא שילוב של כל התוצאות החלקיות. אפשר לקבוע סוגריים מוכלים בתוך סוגריים. בכל מקרה יבוצעו הסוגריים הפנימיים ביותר לראשונה. הערכת ביטוי בתוך סוגריים נעשית לפי כללי הערכת ביטויים חשבוניים, ותחשב כתוצאה חלקית לצורך המשך העיבוד.

נחזור לדוגמה הקודמת כדי להדגים את השימוש בסוגריים. נוסיף לביטוי שבדוגמה הקודמת סוגריים ועתה יכתב כך:

-Y**4+((K*X*B/D)**2-4*C1)/(K*X);

סדר הערכת הביטוי יהיה:

1) Y**4- יבוצע ראשון והתוצאה תשמר ב-t1.
2) הביטוי החלקי בסוגריים הראשון משמאל יסרק כדי להחליט על סדר ההערכה. בתוך סוגריים אלו ישנו ביטוי חלקי נוסף, גם הוא בסוגריים ולכן הוא יוערך בתחילה. תוצאת הביטוי K*X*B/D תשמר ב-t2.
3) הפעולה הבאה שתתבצע היא t2**2, התוצאה היא t3.
4) 4 יוכפל ב-C1 והתוצאה תשמר ב-t4.
5) t4 יחוסר מ-t3 והתוצאה תשמר ב-t5. בשלב זה סיימנו את הטיפול בחלק הביטוי הראשון, אשר ערכו שמור עכשיו ב-t5.
6) חלק הביטוי הבא הוא K*X, מכפלה זו תתבצע והתוצאה תשמר ב-t6.
7) צורת הביטוי כעת היא t1 +t5/t6. תחילה יבוצע t5/t6 והתוצאה תתוסף ל-t1 ותועבר למשתנה המכיל את התוצאה.

בביטויים החשבוניים אפשר להשתמש בסוג אחד של משתנים או במשתנים מסוגים שונים. שימוש במשתנים מסוג אחד מפשט את התוכנית ומעלה את אמינותה. מאידך, קיימים מקרים בהם חובה להשתמש בביטוי אחד במשתנים מסוגים שונים. המהדיר מזהה עובדה זו ועורך "הסבה" בין סוגי הביטויים השונים בזמן ההידור או בזמן ביצוע התוכנית. ההסבה תבוצע לפי תכונות המשתנה המקבל את התוצאה.

פסוקי ההשמה

העברת נתונים בזכרון המחשב נעשית ע"י השמה (assignment) של ערך ביטוי (expression) למשתנה מסויים. הביטוי שאת ערכו מעבירים למשתנה יכול להיות קבוע, משתנה או ביטוי המייצג פעולות חשבוניות.

פסוק ההשמה - assignment statement - הוא הפסוק העיקרי המשמש למטרת העברת נתונים, והוא מאופיין ע"י סימן ההשמה (=) ואינו מכיל מלת-מפתח כלשהיא. משמאל לסימן ההשמה מותר לרשום <u>רק משתנה</u>. שימושים של פסוק ההשמה:

א. העברת ערך של קבוע מספרי למשתנה ;PRICE = 57.3.
ב. העברת ערך של קבוע מחרוזת: ;'CODE = 'XYZ
ג. העברת ערכו של משתנה IN1 למשתנה SUM ;SUM = IN1. המשתנה IN1 נשאר ללא שינוי.
ד. העברת ערך ביטוי: ;TOTAL =(T1+T2)/3. הערך של תוצאת החישוב של הביטוי החש-בוני המופיע בצד ימין של סימן ההשמה מועבר למשתנה TOTAL.

אפשר להשתמש בפסוק השמה אחד בכדי לבצע העברת נתונים ליותר ממשתנה אחד,

לדוגמה,

```
X1, W_AX = MØNE;
```

ערכו של MONE יועבר לשני המשתנים הרשומים משמאל לסימן ההשמה, וזאת כאילו היינו כותבים:

```
X1 = MØNE;
W_AX = MØNE;
```

תכונות הביטוי שמימין עשויות להיות שונות מאלו של המשתנה שמשמאל לסימן ההשמה. לכן, עשוי פסוק ההשמה לשמש גם למטרות הסבה ועריכה.

<u>תרגיל:</u>

יש לחשב את הערך של:

$$RESULT = \frac{5 * FACTØR - (A+6)}{VALUE}$$

נתונים ערכי המשתנים: FACTOR שוה ל-22, A שוה ל-18, VALUE שוה ל-2.

התוכנית:

```
1   FACTØR = 22;
2   A = 18;
3   VALUE = 2;
4   RESULT = (5 * FACTØR - ( A+6))/VALUE;
```

<u>הסבר:</u>

שורות 1, 2, 3 הן פסוקי השמה המקנים לשלושת המשתנים את הערכים לפי תנאי הבעיה. בשורה 4 ישנו פסוק השמה המכיל ביטוי חשבוני. שים לב לצורת הצגת הבי- טוי החשבוני בתוכנית, ולסוגריים התוחמים את המונה כולו לצורך חילוק במכנה. בסיומו של כל פסוק ישנו המפריד נקודה-פסיק. כתיבת הפסוקים חופשית ורווחים יכולים להופיע בכל מקום שהוא בין מזהים ומפרידים.

קלט פלט בסיסי

בשם הוראות קלט/פלט (בקיצור - ק/פ) מכנים את ההוראות הגורמות את מעבר הנתונים בין אמצעי אחסנה (של נתונים) חיצוניים כגון, סרטים מגנטיים, תקליטים מגנטיים, כרטיסים מנוקבים, נייר ומסופים לבין זכרון המחשב. יש ב-PL/I מגוון רחב של הוראות ק/פ והשימוש בהן מותנה ביישום בו משתמשים.

ביישומים מדעיים המטפלים בכמויות קטנות של נתונים במבנה פשוט נשתמש בהו-ראות מסוג Stream Oriented I/O. ביישומים ניהוליים, בהם גדולה בדרך כלל מור-כבות הקבצים ויש חשיבות רבה מאד לצורת ארגון הנתונים ולאופן אחזורם, נשתמש בעיקר בהוראות מסוג Record Oriented I/O. נדון בתחילה בהוראות Stream Oriented I/O כיון שהן פשוטות יותר והשימוש בהן איננו דורש הכנות רבות.

הוראות קלט/פלט בשיטת Stream מבצעות, בנוסף לפעולה העיקרית שלהן, גם פעו-לת הסבה של נתונים מספריים. בקלט, מוסבים נתונים מספריים לפי תכונות המשתנים המפורטים בהוראה. בפלט, מוסבים המספרים מהצורה שבה הם מוצגים במשתנים אל מח-רוזת תוים הניתנת להדפסה. מחרוזות תוים נשארות ללא שינוי (מחרוזת תוים מוג-בלת בגרשים).

קיימות שלוש צורות להוראות ק/פ מסוג stream:

א. העברת נתונים לפי שם-משתנה Data-Directed Input/Output.
שיטה זו מיועדת להעברת מספר נתונים קטן, כאשר כל נתון מופיע בשמו ובערכו.
ב. העברת רשימות נתונים List-Directed Input/Output.
בשיטה זו אפשר לבצע קריאה וכתיבה של נתונים רבים, ברשימות ערכים בלבד.
ג. העברת נתונים במבנה Edit-Directed Input/Output.
שיטת ק/פ זו מאפשרת שליטה על מבנה הנתונים בקלט ובפלט, בכל הוראה והוראה.

כל הוראה צריכה לכלול את הפרטים הבאים:

א. שם הקובץ של הקלט או קובץ הפלט.
בדוגמאות הבאות נשתמש בקבצים הסטנדרטיים של מערכת ההפעלה לקלט ופלט, קורא-הכרטיסים והמדפסת, אשר אין צורך להגדירם במפורש בהוראה. המהדיר בוחר בהם כברירת-מחדל.
ב. רשימת המשתנים אליהם יועברו הנתונים מקובץ הקלט או שמהם יועברו הנתונים אל קובץ הפלט. תכונות המשתנים מוגדרות בנפרד.
בהוראות מסוג Data-Directed אפשר לותר על רשימה זו, כפי שיוסבר בהמשך.
ג. מבנה הנתונים בקלט או בפלט. פרמטר זה דרוש רק בהוראות מסוג Edit-Directed.

בפרק זה נדון בהוראות מסוג Data ו-List בלבד

קלט/פלט בשיטת DATA

הוראות קלט פלט בשיטת Data (Data-Directed I/O) מאפשרות העברת נתונים לפי שם משתנה. רשימת הנתונים בקלט ורשימת הנתונים בפלט כוללת את שם המשתנה ובצמוד לו, במבנה של פסוק השמה, את ערכו של המשתנה. ערך המשתנה רשום כקבוע.

ק ל ט

המבנה הכללי של הוראת קלט מן הסוג הנ"ל הוא כדלקמן:

GET DATA (רשימת משתנים);

דוגמה להוראת קלט:

GET DATA (NAME, CASH, INVENTØRY);

הוראה זו תגרום לקריאת שלושה נתונים מתוך קובץ קלט סטנדרטי של מערכת ההפעלה אשר מוגדר בדרך כלל בכרטיסים מנוקבים.

כאשר רשימת המשתנים אינה כלולה בהוראת GET, תבוצע קריאה של כל המשתנים הנמצאים ברשימת הנתונים בקלט, כפי שיוסבר להלן.

הנתונים יופיעו בכרטיס המנוקב במבנה הבא:

NAME='ABCD', CASH=34593, INVENTORY=1753.22;

דהיינו, הנתונים מנוקבים במבנה של פסוקי השמה המבנה חופשי למדי וחלים עליו כללי הכתיבה הבאים:

א. בין נתון לנתון ברשימת הנתונים יבוא מרווח אחד לפחות ו/או פסיק.

ב. שמות המשתנים בהוראת הקלט וברשימת הנתונים בכרטיס צריכים להיות זהים. שמות שאינם זהים לא ישתתפו בפעולת הקלט, כלומר נתונים אלו לא יקראו לזכרון המחשב. אולם כאשר ההוראה אינה כוללת שמות-משתנים, תתבצע קריאה של כל הנתונים שברשימה.

ג. בסוף רשימת הנתונים המתאימה להוראת קלט אחת יבוא המפריד (;). הרשימה יכולה לכלול יותר מכרטיס אחד.

ד. סדר הופעת הנתונים ברשימה, אינו חייב להיות זהה לסדר כתיבת המשתנים בהוראה

פ ל ט

המבנה הכללי של הוראת פלט מן הסוג הנ"ל הוא כדלקמן:

PUT DATA (רשימת משתנים);

דוגמה להוראת פלט:

PUT DATA (ITEM_NAME, PRICE, VALUE);

הוראה זו תגרום לכתיבת שלושת הנתונים הנ"ל על קובץ הפלט הסטנדרטי של מערכת ההפעלה המוגדרת בדרך כלל כקובץ הדפסה במדפסת. הנתונים יודפסו ברשימה במבנה הבא, לדוגמה:

ITEM_NAME='SHEM PARIT' PRICE=20.5 VALUE=2562.5;

שם המשתנה וערכו יודפסו במבנה של פסוקי השמה, אשר דומה למבנה רשימת הנתונים בקלט. המשתנים וערכיהם מודפסים בזה לאחר זה בשורת הדפסה אחת או יותר, בטורים קבועים: 1, 25, 49, 23, 97, 121. בסוף רשימת הנתונים המודפסים מופיע המפריד "נקודה-פסיק".

קלט/פלט בשיטת LIST

הוראות קלט/פלט בשיטת LIST (List-Directed I/O) מאפשרות העברת נתונים לפי רשימות של ערכים. הוראות אלו דומות להוראות קלט/פלט בשיטת Data. ההבדל הוא בכך שכאן מופיעים הנתונים ברשימות הקלט והפלט בערכיהם בלבד ללא שמות המשתנים. שמות המשתנים נמצאים בהוראות התוכנית בלבד.

בשיטת List-Directed יכולים להופיע ברשימות הקלט והפלט קבועים מסוג מספרים או מחרוזות תוים כמו בשיטת Data-Directed. בשיטת List אפשר לציין רשימת המשתנים בהוראה גם ביטויים חשבוניים, כפי שיוסבר בהמשך.

המבנה הכללי של הוראות ק/פ בשיטה הנ"ל הוא כדלקמן:

GET LIST (רשימת משתנים);

PUT LIST (רשימת משתנים);

ק ל ט

"רשימת המשתנים" בהוראת GET היא רשימה של משתנים שאליהם יש לבצע השמה של הערכים המופיעים ברשימת הקלט בקובץ. המשתנים ברשימה מופרדים האחד מן השני ע"י פסיק (,), כבדוגמה הבאה:

GET LIST (NUMBER, QUANTITY, CODE);

הוראה זו תגרום לקריאת שלושה נתונים מקובץ הקלט והשמתם במשתנים שברשימה בסדר בו נרשמו. דהיינו, הנתון הראשון בקלט יושם ב-NUMBER השני ב-QUANTITY

והשלישי ב-CODE. ביצוע הוראת הקלט יסתיים בנקודה זו, כיון ש"רשימת-המשתנים" תמה.

אם רשימת הנתונים בקובץ הקלט תהיה במבנה הבא ....,'ABC' ,213 ,9532
ואז לאחר ביצוע הוראת הקלט, יהיה תוכן המשתנה NUMBER-9532, תוכנו של QUANTITY יהיה 213, וזה של CODE יהיה 'ABC'.

מבנה הנתונים בקלט בשיטת LIST שונה ממבנה הקלט בשיטת DATA. באחרון הקלט הוא במבנה של פסוקי השמה דהיינו: שם המשתנה, סימן ההשמה וערך המשתנה. לכן, אין גם חשיבות להתאמה בין סדר רשימת הנתונים בהוראת הקלט לבין סדר הנתונים בקלט עצמו. בשיטת List-Directed I/O לא מוזכר שם המשתנה בקובץ הקלט ולכן ההתאמה נעשית לפי סדר רשימת המשתנים והערכים שבקלט. בין נתון לנתון בקלט יכול לבוא המפריד פסיק (,) או מרווח אחד לפחות.

פ ל ט

רשימת הנתונים ברשימת הוראת הפלט נבדלת מזו של הוראת הקלט בכך ש"הנתון" יכול לבוא בכמה צורות: קבוע, משתנה, ביטוי. הפלט מכיל תמיד ערך סופי ולכן, אם "הנתון" הוא קבוע יכתב ערכו של הקבוע, אם "הנתון" הוא משתנה - יכתב ערכו הנוכחי של המשתנה, ואם "הנתון" הוא ביטוי (כלשהו), ערכו יחושב והתוצאה תועבר לפלט. דוגמה:

```
PUT LIST (X, 122.25, (Y-A)*3);
```

הוראה זו תגרום להדפסת שלושה נתונים: הנתון הראשון יהיה תוכנו הנוכחי של המש-תנה X, הנתון השני שיכתב יהיה 122.25, והנתון השלישי שיכתב יהיה תוצאת הביטוי החשבוני 3*(Y-A). ההבדל השני, אם כך, בין List-Directed I/O לבין Data-Directed I/O הוא בכך שהאחרון מגביל את רשימת הפלט למשתנים בלבד.

הנתונים יכתבו בקובץ הפלט במקומות קבועים בשורה, כמו ב-Data-Directed I/O אולם בהבדל נוסף. כאן יופיע רק תוכן הנתון ללא שמו.

תרגיל

יש לחשב ממוצע של 4 מספרים הנתונים בכרטיס מנוקב. השתמש בשיטת Data-Directed I/O.

הקלט: N=100 M=20 L=60 K=40;

התוכנית: בשלב זה נניח שהמשתנים יופיעו בזכרון עם תכונות לפי ברירת המחדל,

לכן אפשר להמנע מהגדרה מפורשת שלהם.

```
AVERAGE: PRØCEDURE ØPTIØNS (MAIN);
         GET DATA (N,M,L,K);
         AVR=(N+M+L+K)/4;
         PUT DATA (AVR);
         END;
```

<u>הסבר:</u>

א. הפסוק הראשון בתוכנית הוא פסוק PROCEDURE, שבו מצויין משמאל שם התוכנית AVERAGE, ומימין - רשימת פרמטרים אשר במקרה זה מכילה את מלת המפתח MAIN לציון החלק הראשון של התוכנית אשר מקבל את הפיקוח ממערכת ההפעלה. התוכנית מסתיימת בפסוק END.

ב. הוראה הקלט GET תקרא את הנתונים שבקלט, בהתאם לרשימה שבסוגריים. הוראת הפלט PUT תדפיס את התוצאה AVR.

ג. הוראת החישוב היא פסוק השמה המכיל ביטוי חשבוני. בביטוי החשבוני השתמשנו בסוגריים כדי לציין שפעולת החיבור תבוצע לפני פעולת החילוק.

<u>הפלט:</u> AVR=54;

שים לב שבסוף רשימות הנתונים בקלט ובפלט מופיע המפריד "נקודה-פסיק".

תרגיל מסכם

מטרת התרגיל לסכם את נושא פסוקי ההשמה והוראות קלט ופלט בשיטות Data-Directed ו-List-Directed.

<u>הבעיה:</u>

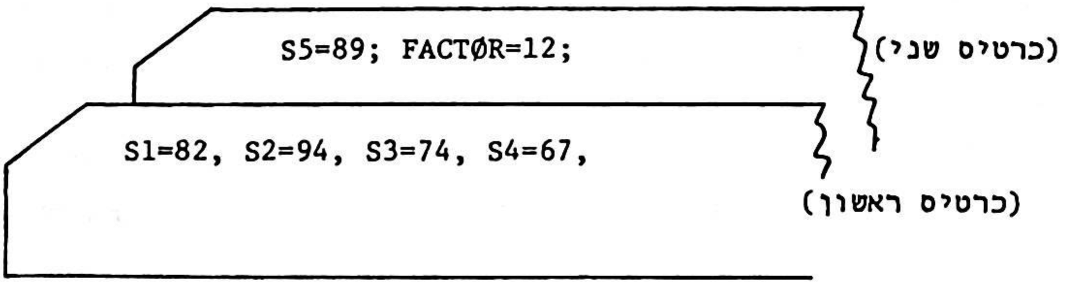
יש לקרוא ציונים של תלמידים המנוקבים בכרטיסים, לחשב את הממוצע המשוקלל ולהדפיס את התוצאה. הקלט והפלט במבנה DATA.

<u>תאור התוכנית:</u>

התוכנית קוראת 5 ציונים ב-5 מקצועות לימוד, מחשבת את ממוצע הציונים ומדפיסה את חמשת הציונים ואת הממוצע. לאחר מכן קוראת התוכנית מספר נוסף (גורם שיקלול), מוסיפה אותו לציון הראשון והשלישי, מחשבת את הממוצע המשוקלל ומדפיסה שוב את חמשת הציונים ואת הממוצע.

<u>הקלט</u> לתוכנית ינוקב בשני כרטיסים כמפורט כאן. את הנתונים אפשר לנקב בכל מקום שהוא בכרטיס. יש לשמור רק על ההפרדה בין נתון אחד לשני בפסיק ו/או

במרווח, ובסיום הרשימה (כאן בכרטיס השני) צריך להופיע המפריד נקודה-פסיק.



<u>פלט</u> התוכנית יופיע במספר שורות, ברציפות. בסיום כל רשימת משתנים הכלולה בהו-ראת PUT אחת יופיע המפריד נקודה-פסיק.

```
S1=82          S2=94          S3=74          S4=67          S5=89
MEAN=81.2;     S1=94          S2=94          S3=86          S4=67
S5=89          MEAN=85;
```

<u>התוכנית:</u>

PL/I Coding Form

TALMIDIM - MARKS

A.B.   DATE SEPT 19[illegible]

PL/I STATEMENT

```
TALMIDIM: PRØCEDURE ØPTIØNS(MAIN);
          DCL (S1,S2,S3,S4,S5) FIXED DECIMAL (3),
              MEAN FIXED DECIMAL (4,1),
              FACTØR FIXED DECIMAL (2);
          GET DATA (S1,S2,S3,S4,S5);
              MEAN = (S1+S2+S3+S4+S5)/5 ;
          PUT DATA (S1,S2,S3,S4,S5,MEAN) ;
          GET DATA (FACTØR);
              S1 = S1 + FACTØR ;
              S3 = S3 + FACTØR ;
              MEAN = (S1+S2+S3+S4+S5)/5 ;
          PUT DATA (S1,S2,S3,S4,S5,MEAN) ;
      STOP;
END TALMIDIM;
```

הסבר לתוכנית

התוכנית נכתבת בטופס בן 80 טורים במבנה חופשי, בטורים 2 עד 72. טור מס' 1 אינו בשימוש ב-PL/I. הטורים 73 עד 80 מוקדשים לסימון זהוי התוכנית: בטורים 73 עד 77 נרשם קוד התוכנית TALMD ובטורים 78 עד 80 נרשם מספר סדורי של השורות שנכתבו. שים לב שהמספרים חייבים להיות בסדר עולה אולם רצוי שלא יהיה עוקבים (כדי לאפשר תוספות בין השורות). טורים 73 עד 80 אינם נבדקים ע"י המהדיר, ואינם שייכים לתוכנית.

הפסוק הראשון בתוכנית PL/I הוא PROCEDURE. כאן כולל הפסוק את שם התוכנית TALMIDIM ואת הפרמטר MAIN המציין שזהו הקטע הראשון בתוכנית.

ההגדרה של המשתנים בתוכנית נעשית בשורה 30, בעזרת הפסוקDECLARE או בקי-צור - DCL. המשתנים S1 עד S5 הם באותו מבנה ואופי ולכן יוגדרו במקובץ, כפי שמתואר בשורה 30. בשורה 50 מוגדר המשתנה MEAN (עבור הממוצע): אורך השדה 4 ספרות, הספרה הימנית היא ספרה עשרונית. כל המשתנים הםFIXED DECIMAL. ההגדרה של כל המשתנים יכולה להעשות בפסוק אחד אשר משתרע על מספר שורות, בין הגדרה אחת לשניה רושמים פסיק ובסוף הפסוק נקודה-פסיק.

בשורה 90 רשומה הוראת הקלט של 5 משתנים. שים לב שלמרות שהם מנוקבים בשני ׳רטיסים אין זה משנה לגבי התוכנית. היא תקרא את הכרטיסים עד לקבלת המפריד נקו-ׄ-פסיק. חישוב הממוצע הפשוט נעשה בשורה 110. שים לב לסוגריים הקובעים שפעו-׳ת החיבור תתבצענה לראשונה ורק אח"כ - פעולת החילוק. הוראת הפלט אחת בשורה 13 תדפיס את 5 המשתנים ואת הממוצע שחושב. שים לב שההדפסה תבוצע במספר שורות, עד להשלמת כל המשתנים שברשימה.

בשורה 150 ישנה הוראת קלט של המשתנה FACTOR. מכיון שהכרטיס כבר בזכרון (ע"י פעולת קריאה קודמת) תשמש הוראה זו להעברת הנתון אל התוכנית. בשורות 170, 180 מחברים את גורם השיקלול לציונים S1, S3. בשורה 200 מחשבים את הממוצע המשוק-לל, ולאחר מכן מדפיסים את התוצאה.

בסיום ביצוע התוכנית יש להעביר את הפיקוח למערכת ההפעלה. לצורך זה משתמשים בהוראה STOP (שורה 230). כאשר זוהי ההוראה האחרונה בתוכנית, לפני ההוראה END, אפשר להשמיט אותה.

ההוראה האחרונה הנכתבת בתוכנית היא פסוק END, אשר אפשר להוסיף לו מימין את שם התוכנית, כפי שמופיע בתרגיל.

הערה:

אם רוצים להשתמש לקלט ופלט בשיטת LIST, אפשר להחליף את מלת המפתח DATA שבהוראות, במלה LIST. הקלט והפלט יהיו ללא שמות המשתנים, אלא יופיעו רק הערכים המספריים שלהם. בנוסף, אפשר לרשום במקום המשתנהMEAN שבהוראת PUT את כל הביטוי החשבוני.

פרק 4: הוראות פיקוח

מ ב ו א

התוכנית מתבצעת כרגיל לפי סדר הפסוקים המרכיבים אותה. הצורך לשינוי סדר הביצוע קיים כאשר נוצרים תנאים בהם יש להפסיק ביצוע רצף הפקודות ולעבור לקטע תוכנית רצוי, לדלג על קטע תוכנית, לחזור ולבצע קטע תוכנית לפי התניות כלשהן וכד'. שינוי רצף זרימת הביצוע נעשה ע"י הוראות פיקוח. הוראות הפיקוח הבסיסיות ב-PL/I הן: GO TO , IF , DO ובהן נדון בפרק זה.

הוראות הסתעפות בלתי מותנית

הסתעפות לכתובת קבועה

הוראת GO TO (או GOTO, ללא רווח בין המלים) גורמת להסתעפות בלתי מותנית אל פסוק מסוים בתוכנית. המבנה הכללי של ההוראה הוא:

GØ TØ (תוית);

ה"תוית" חייבת להיות שם של תוית מוכרת בתוכנית, אחרת אין שום משמעות להוראה.
דוגמאות:

```
GØ TØ L1;
GØTØ FINISH;
GØTØ GØTØ;
```

הפסוק השלישי תקף בתנאי שבתוכנית יופיע המזהה GOTO כתוית של אחד הפסוקים.
הערה: בדרך כלל לא נוהגים להשתמש בשם שהוא חלק מהשפה למטרת תויות או משתנים, אולם כאן זוהי דוגמה נוספת לגמישות הרבה בתכנות בשפה זו.
דוגמה:

```
AGAIN: GET DATA (PRICE, QUANT);
       VALUE=PRICE*QUANT;
       ------
       PUT DATA (VALUE);
       GØ TØ AGAIN;
NEXT: -------
```

בדוגמה זו תשנה הוראת GO TO את סדר ביצוע התוכנית: לאחר הוראת PUT לא תתבצע ההוראה בתוית NEXT, אלא הפיקוח יועבר להוראה AGAIN. הסתעפות זו תתבצע בכל

פעם שביצוע התוכנית יגיע להוראת GO TO.

הסתעפות לפי משתנה-תוית

הוראת GOTO שתוארה מייצגת הסתעפות לתוית (כתובת) קבועה. קיימים מצבים בהם נדרש לבצע במהלך התוכנית הסתעפות לתויות שונות ע"י אותה הוראה GOTO. הס- תעפות כזו נעשית ע"י השימוש ב"משתנה-תוית".
הצורה הכללית של הוראת הסיעוף היא:

GØ TØ (משתנה-תוית);

הגדרת "משתנה-התוית" תעשה בפסוק DECLARE והיא יכולה (ברשות) לציין את כל שמות- התויות שאת ערכיהם יכול המשתנה לקבל במהלך התוכנית, לדוגמה:

DECLARE (משתנה-תוית) [תוית, תוית.....] LABEL;

כאשר רוצים שמשתנה-התוית יכיל ערך (כתובת) של תוית כלשהיא, יש להשתמש בפסוק השמה,

(משתנה-תוית) = (תוית);

ז השימוש במשתנה-תוית נדגים בעזרת הדוגמה הבאה:

```
      DECLARE TRANS (L1, L2, NEXT, ERR) LABEL;
      TRANS = L2;
GET:  GET DATA (A, B, C);
      -------
      GØ TØ TRANS;
      -------
L2:   -------
      TRANS = ERR;
      GØ TØ GET;
      -------
L1:   -------
ERR:  -------
```

הסבר: המשתנה TRANS מוגדר כמשתנה-תוית. אפשר להגביל את הערכים שאפשר להעביר למשתנה-תוית בזמן ביצוע התוכנית באמצעות פסוק ההצהרה. אופציה זו איננה חובה, אבל השימוש בה עשוי לייעל את התוכנית במונחים של זמן ביצוע, אך בעיקר - לשמש תעוד לתוכניתן על כוונותיו בתוכנית.

בדוגמה הגבלנו את ערכי TRANS האפשריים ל-NEXT, L2, L1 ו-ERR. בכל מקרה, הערכים המועברים ל-TRANS חייבים להיות מסוג תוית.

בפסוק השני יקבל TRANS את הערך L2 ולכן, בפסוק 5 תתבצע הסתעפות לפסוק 7, בעל תוית L2. בפסוק 8 מועבר ערך חדש ל-TRANS - התוית ERR. לאחר ביצוע הפסוקים 3, 4 תתבצע הסתעפות נוספת, והפעם לפסוק 12, הנושא את התוית ERR, מכיון שמשתנה התוית TRANS מכיל את ערך התוית ERR.

בדוגמה זו הושמטה כל הלוגיקה אשר כוונה את ההחלטה להקנות למשתנה-התוית ערך זה או אחר. למשל, במצב של שגיאה בתוכנית ינתן למשתנה-התוית TRANS הערך ERR כדי שיבוצע סיעוף לקטע התוכנית המטפל במצבי שגיאה.

הוראות התנייה

לעתים קרובות יש צורך לבצע פסוק אחד או קבוצת פסוקים רק כאשר תנאי מסויים מתקיים. כלומר, יש להסתעף מרצף ההוראות לאחר הערכת ביטוי כלשהוא. אם התנאי יתקיים יבוצעו הפסוק או הפסוקים במסלול אחד של התוכנית, ואם התנאי איננו מתקיים - תמשיך התוכנית בביצוע מסלול אחר של הוראות.

הוראת ההתנייה IF מבצעת את הערכת הביטוי המשמש כתנאי ובהתאם לתוצאות ההערכה ממשיכה בביצוע ההוראות.

בפסוק ההתנייה IF אפשר לכלול ביטויי השוואה וביטויים לוגיים שונים. תוצאת ההערכה של ביטויים אלו עשויה להיות התשובה "כן" או התשובה "לא", או בשפה אחרת, הביטוי עשוי להיות "נכון" (true) כאשר התנאי מתקיים ו"לא-נכון" (false) כאשר התנאי איננו מתקיים. התשובה קובעת את המשך מסלול הביצוע של התוכנית.

למשל: נתונה רשימת תלמידים וציוניהם. לכל תלמיד מחושב ממוצע הציונים ונבדק מול ערך קבוע (הציון "עובר"). אם הממוצע גבוה מציון "עובר" - רושמים את התלמיד ברשימת התלמידים שעוברים לשלב הלימוד הבא, ואם ציונו הממוצע נמוך מציון "עובר" - מדפיסים עבורו מכתב הודעה והזמנה לבחינה בע"פ.

ביטויי השואה

פעולות השואה מציגות יחס השואה בין גורמים. על פעולות המציגות יחס לוגי בין גורמים נעמוד בהמשך. פעולות השואה האפשריות הן:

גדול מ-	>	לא שוה ל-	¬=
לא גדול מ-	¬>	קטן מ-	<
גדול מ-, או שוה ל-	>=	לא קטן מ-	¬<
שוה ל-	=	קטן מ-, או שוה ל-	<=

לבניית ביטויי השוואה הכוללים פעולות אלו אפשר להשתמש במשתנים וקבועים. הצורה הכללית של ביטוי ההשוואה היא:

{משתנה / או קבוע} (גורם היחס) {משתנה / או קבוע}

לדוגמה:

```
QUANTITY >= 15
```

בביטוי זה מושווה ערכו של המשתנה QUANTITY ל-15. אם ערכו של QUANTITY גדול או שוה ל-15 הביטוי יהיה "נכון", אחרת - "לא נכון".

```
(A+B)**(K/5)<= X-Y**2
```

בדוגמה זו יקבע ערכו של הביטוי החשבוני שמשמאל לגורם היחס, אח"כ יקבע ערכו של הביטוי החשבוני מימין לגורם היחס, ולבסוף יושוו שתי תוצאות הביניים זו לזו.

ההשואה של נתונים מספריים נעשית בדרך אלגברית. ערכים שליליים קטנים מערכים חיוביים.

ביטויים לוגיים

לביטויים שהצגנו בסעיף הקודם ואשר ערכם עשוי להיות "נכון" או "לא-נכון", נוהגים לקרוא גם בשם "ביטויים לוגיים". תוצאת ביטוי לוגי מאוחסנת בזכרון המחשב כתוצאת ביניים במהלך הערכה של ביטוי מורכב, או כתוצאה סופית של ביטוי. תוצאת הביטוי הלוגי מאוחסנת בסוג של נתונים הנקרא נתונים לוגיים.

נתון לוגי נקרא ב-PL/I בשם מחרוזת סביות (bit string). מחרוזת סביות מוגדרת כרצף של סביות אשר כל אחת מהן יכולה להיות במצב של 1 או 0.

תוצאה של ביטוי לוגי נשמרת בנתון מסוג BIT(1). פעולה לוגית שערכה "נכון", תיוצג ע"י '1'B, פעולה לוגית שערכה "לא נכון" תיוצג ע"י '0'B.

דוגמה:

```
DECLARE B BIT(1), L BIT(1);
B = X + Y > Z**2;
L = M = N;
```

המשתנה B בפסוק 2 יקבל את הערך B'1' אם X+Y גדול מ-Z^2, או את הערך B'0' כאשר התנאי אינו מתקיים, כלומר X+Y אינו גדול מ-Z^2.
המשתנה L בפסוק 3 יקבל את הערך B'1' אם M=N, או את הערך B'0' כאשר M אינו שוה ל-N. סימן = הראשון משמאל בפסוק 3 הוא סימן ההשמה, והסימן = השני משמאל הוא סימן "שוה ל-". כדי להקל על "קריאות" התוכנית רצוי היה לכתוב את פסוק 3 כדלקמן: L=(M=N). דוגמה זו ממחישה את העובדה שלמרות החופש הרב בכתיבה יש לנהוג במשמעת כלשהיא כדי להעלות את בהירות והבנת התוכנית.

<u>פעולות במחרוזות סביות</u>

ביטוי לוגי הינו רצף של מחרוזות סביות מסוג משתנים או קבועים המופרדים ע"י פעולות של אופרטורים לפעולות לוגיות (בוליאניים):

¬ לא, not
& וגם, and
| או, or (קו זקוף)

האופרטורים "או", "וגם" הם מסוג infix, כלומר פועלים על צמד מחרוזות. האו-פרטור "לא" הוא prefix, כלומר פועל על מחרוזת בודדת.

תוצאה של פעולה בין שתי מחרוזות היא מחרוזת סביות באורך המחרוזות המש-תתפות בפעולה. כאשר המחרוזות שוות בארכן, יהיה אורך מחרוזת התוצאה כאורך מחרוזות אלו. כאשר המחרוזות שונות באורכן - יתוספו (זמנית לצורך הפעולה) למחרוזת הקצרה סביות 0 מימין. מחרוזת התוצאה תהיה באורך המחרוזת הארוכה שביניהן. כאשר בבטוי ישנן מחרוזות אחדות תבוצע הפעולה בין שתי מחרוזות (לפי סדר עדיפות שנדון בו בהמשך) ומחרוזת התוצאה תופעל עם המחרוזת הבאה, וכן הלאה.

פעולות-מחרוזת מבוצעות על המחרוזת או המחרוזות המשתתפות סבית אחר סבית משמאל לימין. כל סבית במחרוזת התוצאה מקבלת ערך על פי הטבלה הבאה:

A "או" B A | B	A "וגם" B A & B	"לא" B ¬B	"לא" A ¬A	סבית במחרוזת B	סבית במחרוזת A
1	1	0	0	1	1
1	0	1	0	0	1
1	0	0	1	1	0
0	0	1	1	0	0

<u>דוגמה:</u>

נגדיר את המחרוזות הבאות:

```
DECLARE RSLT BIT(6), MASK BIT(6),
        CØDE1 BIT(6), CØDE2 BIT(6);
```

נציב את הערכים הבאים למשתנים:

```
MASK = '011010'B;
CØDE1= '101111'B;
CØDE2= '001011'B;
```

RSLT הוא המשתנה שיכיל את מחרוזת התוצאה של הביטויים הבאים:

א. RSLT = ¬MASK

"לא" משמעותו "היפוך" הסביות של המשתנה MASK, ולכן ערכו של RSLT יהיה '100101'B.

ב. RSLT = CØDE1 & CØDE2

מחרוזת התוצאה תכלול כתשובה "נכון" כאשר <u>באותו</u> מקום יחסי <u>בשתי</u> המחרוזות הנבד-קות ישנה הספרה 1.

מחרוזת CODE1 מכילה	101111
מחרוזת CODE2 מכילה	<u>001011</u>
ולכן מחרוזת התוצאה RSLT תהיה	001011

ג. RSLT = CØDE2 | MASK

מחרוזת התוצאה תכלול כתשובה "נכון" את המצבים שבהם ישנה הסבית 1 <u>באחת</u> משתי המחרוזות.

מחרוזת CODE2 מכילה	001011
מחרוזת MASK מכילה	<u>011010</u>
ולכן מחרוזת התוצאה תהיה	011011

ד. ולדוגמה, פעולה עם קבוע: RSLT = MASK & '101'B

מחרוזת הקבוע קצרה ממחרוזת MASK, ולכן יתווספו לו, זמנית, 3 אפסים מימין. הערך הזמני שלו יהיה '101000'B, כלומר מחרוזת בת 6 סביות כאורכה של מחרוזת MASK. התוצאה של פעולה זו תהיה '001000'B.

אם בביטוי מחרוזת יש יותר מאופרטור אחד יש צורך להגדיר את נוהל הערכת הביטוי. הביטוי יוערך משמאל לימין לפי סדר העדיפות של האופרטורים: האופרטור "לא" (¬) הוא בעל העדיפות הגבוהה, אחריו "וגם" (&) ואחרון "או" (|).

השימוש בסוגריים מאפשר שינוי סדר הערכת הביטוי בהתאם לתנאי הבעיה.
בדוגמה הבאה שולבו הסוגריים להצגת סדר העדיפות בהערכת הביטוי:

A|B&C זהה ל- A|(B&C)
A&B|C זהה ל- (A&B)|C
¬A|B&C זהה ל- (¬A)|(B&C)
¬A&¬B|C זהה ל- ((¬A)&(¬B))|C

לסיכום נציג את סדר העדיפות של ביצוע פעולות בביטויים הכוללים אופרטורים חשבוניים, אופרטורים להשואה ואופרטורים לפעולות לוגיות:

העדיפות הגבוהה ביותר   ** prefix + prefix -

* /

infix + infix -

< ¬< <= = ¬= >= > ¬>

&

העדיפות הנמוכה ביותר |

פסוק ההתנייה - IF

בהוראת IF נשתמש כדי לבחור בפעולות אלטרנטיביות על סמך תנאים שונים המתגלים במהלך ביצוע התוכנית.

אפשר להשתמש בהוראת IF באחד משני האופנים הבאים:

1. IF (ביטוי) THEN (יחידה-1);
2. IF (ביטוי) THEN (יחידה-1) ELSE (יחידה-2);

"יחידה" היא פסוק יחיד, קבוצת פסוקים או BEGIN BLOCK. בשלב זה נתיחס רק לאפשרות של פסוק יחיד. קבוצת הפסוקים מוגדרת ע"י פסוק DO, שנדון בו בהמשך. BEGIN BLOCK מהווה יחידה נפרדת בתוכנית, וגם בה נדון בהמשך.

בביצוע פסוק IF נעשית תחילה הערכת "הביטוי". הערכת הביטוי יכולה לתת תוצאה "נכון" המציינת שכל התנאים בביטוי התמלאו, או - תוצאה "לא-נכון" המציינת שתנאי הביטוי לא התמלאו.

כאשר "הביטוי" הוא נכון תבוצע תמיד "יחידה-1" הרשומה לאחר מלת המפתח THEN, ולאחר מכן - הפסוק שרשום לאחר הוראת IF. אם יחידה-1 כללה את ההוראה GOTO - יבוצע סיעוף בהתאם.

כאשר הביטוי הוא לא-נכון, יש הבדל בביצוע בשתי הוראות IF שלמעלה: בהוראה IF הראשונה - יבוצע הפסוק שלאחר פסוק IF. בהוראת IF המורחבת - תבוצע יחידה-2 הרשומה לאחר מלת המפתח ELSE, ולאחר מכן (אם היא לא כללה את ההוראה GOTO) יבוצע הפסוק שלאחר פסוק IF.

את האפשרויות הקיימות אפשר להציג בעזרת התרשימים הבאים:



<u>פסוקי התנייה עם ביטויי השואה</u>

א.

```
DECLARE K FIXED DECIMAL (5);
GET DATA (K);
IF K<0 THEN K = -K;
PUT DATA (K);
```

בדוגמה זו, נבדק ערכו של K. אם K שלילי, תבוצע "יחידה-1" שהיא במקרה זה, פסוק בודד ב-PL/I. בפסוק זה הופכים את סימנו של K, לאחר מכן מודפס ערכו של K. אם K אינו שלילי "יחידה-1" לא תתבצע ורצף הפיקוח יעבור מיד לפסוק 4.

ב.

```
DECLARE (MAX,X,Y) FIXED DECIMAL (5);
GET DATA (X,Y);
IF X<Y THEN MAX=Y;
      ELSE MAX=X;
PUT DATA (MAX,X,Y);
```

בדוגמה זו יש להציב ב-MAX את הערך הגדול מבין שני הערכים ב-X וב-Y. אם X<Y מוצב ערכו של Y ב-MAX, אחרת מוצב ערכו של X ב-MAX. בסיום מודפסים ערכי MAX, X, Y.

ג.

```
IF MASK ¬= '110110'B THEN GØ TØ PRINT;
```

בהוראה זו נבדק ערכו של המשתנה MASK, ואם צרוף הסביות שלו אינו שווה לקבוע הבינרי הנתון - יבוצע סיעוף אל התוית PRINT.

הוראות IF מוכלות

אפשר להכיל הוראות IF בתוך הוראות IF. מצבים אלו נדרשים כאשר באחד מהזרועות של הוראת IF יש צורך לבצע הוראת IF נוספת. שיטה זו אפשרית ואינה מומלצת.

```
IF (ביטוי) THEN IF (ביטוי) THEN (יחידה);
                           ELSE (יחידה);
                ELSE (יחידה);
```

ובדוגמה:

```
1  IF AGE >= 18 & SUG-BRIUT > 70
2     THEN IF WEIGHT > 60
3             THEN GØ TØ ØK;
4             ELSE GØ TØ CHECK;
5     ELSE N= N + 1;
```

בשורה 1 רשומה הוראה IF עם ביטוי מורכב הכולל ביטוי לוגי בעל שני חלקים. הביטוי יהיה "נכון" כאשר AGE יהיה גדול או שוה ל-18 וגם SUG-BRIUT יהיה גדול מ-70. כאשר הביטוי "נכון" יעבור הפיקוח לשורה 2, ואם הביטוי "לא-נכון" יעבור הפיקוח לפסוק ELSE בשורה 5,

בפסוק THEN שבשורה 2, רשומה הוראת IF אחרת הכוללת ביטוי השואה. אם הביטוי "נכון" יבוצע הפסוק THEN בשורה 3, ואם הוא "לא-נכון" יבוצע הפסוק ELSE בשורה 4.

לכאורה, אפשר היה לרשום בהוראת IF אחת את כל התנאים. אולם שים לב שישנם מסלולי פעילות שונים כאשר מתקיימים תנאים שונים: כאשר כל התנאים מת-קיימים עוברים ל-OK. כאשר רק השנים הראשונים (גיל וסוג-בריאות) מתקיימים - עוברים ל-CHECK, וכאשר הם אינם מתקיימים עוברים לשורה 5, ופוסחים על בדיקת המשקל.

פסוקי התנייה עם ביטויים לוגיים

א.

```
IF A & B THEN GØ TØ CHECK;
         ELSE GØ TØ PRINT;
```

כאשר פעולת "וגם" של שני המשתנים A, B נותנת את מחרוזת הכוללת "סביות 1", אחת או יותר, יש לבצע את הסיעוף לתוית CHECK. אם כל סביות התוצאה הן 0, יבוצע סיעוף אל PRINT.

ב.
```
IF CHILDREN  >= 5 & (SALARY  <= 2000 | RØØMS < 2)
    THEN GØ TØ BØNUS;
```

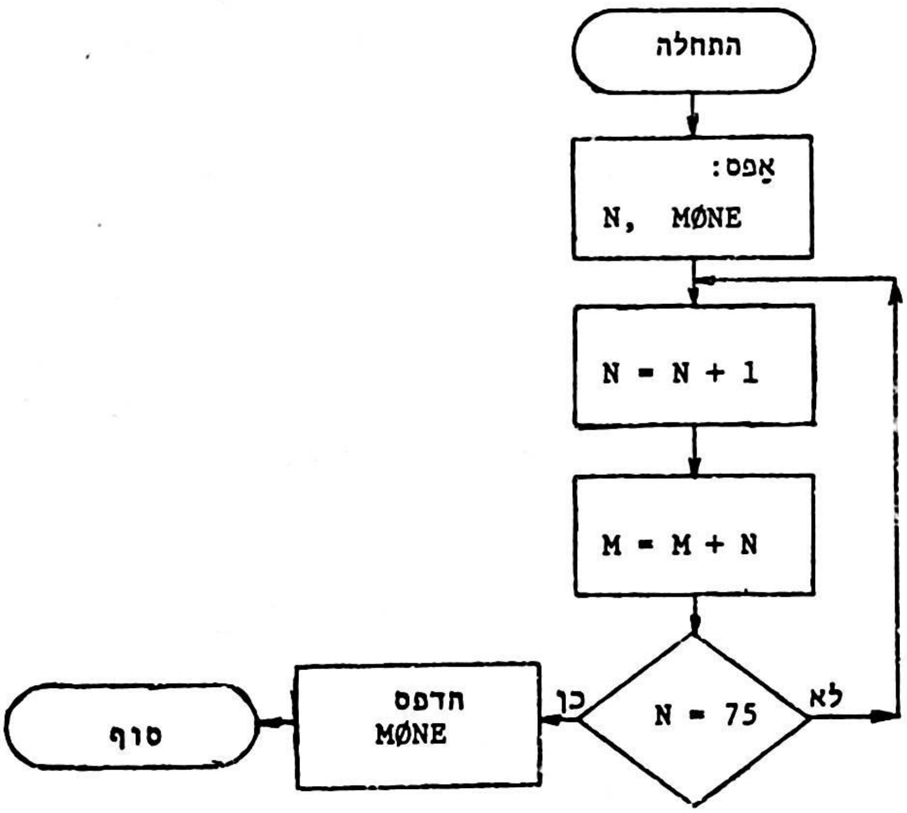
מכיון שהאופרטור "או" (|) הינו בעדיפות נמוכה תחמנו אותו בסוגריים ועל כן פעולת "או" תבוצע לראשונה. אח"כ תבוצע פעולת "וגם" (&) עם הביטוי הראשון שבפסוק. משמעות הפסוק IF היא:

אם מספר הילדים גדול או שוה ל-5 וגם קיים אחד מהתנאים (או) המשכורת קטנה או שוה ל-2000 או מספר החדרים קטן מ-2 יש להסתעף לכתובת BONUS.

כל אחד מביטויי ההשואה מפיק תוצאה בצורת נתון לוגי, סבית 1 או 0 בהתאם לתוצאות ההשואה: התשובה "נכון" מפיקה סבית 1 והתשובה "לא נכון" מפיקה סבית 0. האופרטורים "או", "וגם" פועלים על סביות אלו.

תרגיל

יש לסכם במשתנה MONE את המספרים מ-1 עד 75 ולהדפיס את התוצאה.
תרשים מלבני:



התוכנית:

```
1  CØUNT: PRØCEDURE ØPTIØNS (MAIN);
2         MØNE = 0;
3         N = 0;
4  AGAIN: N = N + 1;
5         MØNE = MØNE + N;
6         IF N ¬= 75  THEN GØ TØ AGAIN;
7         PUT DATA (MØNE);
8         END;
```

<u>הסבר:</u> בתוכנית זו אין הכרזה של נתונים והם נבחרו לפי ברירת המחדל, בהתאם לאות הראשונה של המשתנים - FIXED BINARY. שורות 1 ו-8 הן תחילת וסיום התוכנית. בשורות 2, 3 נקבעים תנאי התחלה - איפוס MONE המשמש לצבירת הסכום ו-N שהוא המספר התור ואשר מסכמים ל-MONE. בפסוק IF בשורה 6 ישנה התנייה שונה מזו שבתרשים, מטעמי נוחות התכנות. את התוצאה מדפיסים בהוראת PUT DATA. אם ההוראה האחרונה המתבצעת היא END אין צורך לרשום לפניה STOP.

<u>תרגיל</u>

יש לחשב שכר עבודה לפי התנאים הבאים:
התעריף לשעת עבודה רגילה הוא RATE. מעל 8 שעות עבודה התעריף הוא ב-50% גבוה יותר.
יש לקרוא מכרטיסים את הערכים מספר-עובד, מספר שעות, תעריף ולהדפיס את מספר-עובד, שכר.
בכרטיס האחרון מכיל השדה מספר-עובד את הערך 9999.

<u>התוכנית:</u>

```
SALARY: PRØCEDURE ØPTIØNS (MAIN);
FIRST : GET LIST (MAN_NØ, HØURS, RATE);
        IF MAN_NØ = 9999 THEN GØ TØ FINISH;
        IF HØURS > 8 THAN PAY = 8*RATE + (HØURS-8)*RATE*1.5;
           ELSE PAY = HØURS * RATE;
        PUT DATA (MAN_NØ, PAY);
        GØ TØ FIRST;
FINISH: STØP;
        END;
```

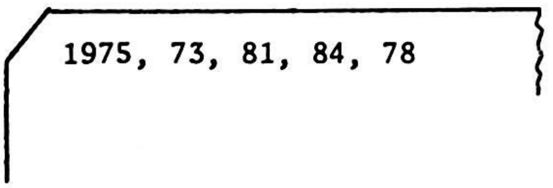
<u>הסבר:</u> מיד לאחר הוראת הקריאה נבדק השדה MAN_NO כדי לדעת אם זהו הכרטיס האחרון, ואז מבוצע סיעוף אל FINISH. אם לא - נערך חישוב השכר בהתאם לתנאי הבעיה. בפסוק IF מוכל ביטוי חשבוני אשר מייצג את החישוב כאשר RATE > 8. בפסוק ELSE כלול החישוב לשעות רגילות.

כדי לפשט את התרגיל לא נכללה כאן הכרזת נתונים.

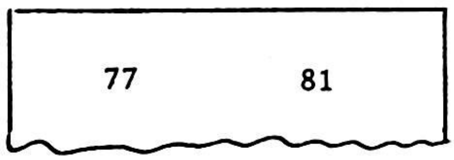
<u>תרגיל</u>

נתון קובץ קלט בכרטיסים אשר מכילים פרטים על תלמידי ביה"ס.
הפרטים הכלולים ברשומת כל תלמיד הם: מספר שנת הלמוד בביה"ס, ציון סימסטר א' באנגלית, ציון סימסטר ב' באנגלית, ציון סימסטר א' במתמטיקה וציון סימסטר ב' במתימטיקה. אם התלמיד בשנתו הראשונה בביה"ס יש לחשב ולהדפיס את ציונו הסופי (שהוא ממוצע בין שני ציוני הסימסטר) רק באנגלית. אם התלמיד בשנתו השניה ואילך יש לחשב ולהדפיס את ציונו הסופי באנגלית ובמתימטיקה.
כדי לפשט נציג את התוכנית לטיפול בכרטיס <u>אחד</u> בלבד.

<u>מבנה כרטיס קלט</u> (בשיטת List-Directed)



<u>מבנה ההדפסה</u>



$$(73 + 81)/2 = 77$$
$$(84 + 78)/2 = 81$$

התוצאה 77 תודפס בטור 25 בשורת ההדפסה, והתוצאה 81 תודפס בטור 49 בשורה.

<u>התוכנית</u>

```
SCØRE: PRØC ØPTIØNS (MAIN);
       DCL YEAR FIXED DECIMAL (4),
           (S1, S2) FIXED DECIMAL (2);
       GET LIST (YEAR, S1, S2);
       PUT LIST ((S1+S2)/2);
           IF YEAR > 1 THEN DØ;
                                GET LIST (S1, S2);
                                PUT LIST ((S1+S2)/2);
                                END;
END SCØRE;
```

הסבר:

פסוקים 1 ו-11 חייבים לבוא בכל תוכנית PL/I. פסוק 1 הוא הפסוק הפותח כל תוכנית ב-PL/I. תוית הפסוק - הכרחית והיא מהווה את שם התוכנית. פסוק 11 הוא פסוק הסיום של התוכנית. כאשר רצף הפקוח מגיע לפסוק הנ"ל, מסתיימת התוכנית, גם אם אין הוראת STOP לפניו.

בפסוק 2 השתמשנו במלת המפתח DCL שהיא השם המקוצר של מלת המפתח DECLARE.

בפסוק 4 קוראת התוכנית שלשה נתונים מהכרטיס: השנה, ושני ציוני האנגלית.
בפסוק 5 מבוצע חישוב הממוצע וההדפסה ע"י הוראת PUT LIST.
בפסוק 7 נבדק התנאי של מספר שנות הלימוד, ואם הוא גדול מ-1 מבוצעת קבוצת הפסוקים הכלולה במסגרת הוראות DO עד END (עד פסוק 9) ואח"כ עוברת התוכנית לפסוק 10.
אם התלמיד בשנתו הראשונה, עובר רצף הפיקוח מיד לפסוק 10 והתוכנית מסתיימת.
אם YEAR > 1, יבוצעו הפסוקים 8 ו-9. שים לב שהקריאה של שני השדות הבאים נעשית מאותו הכרטיס. ההבחנה היחידה לכך היא מספר השדות או הנתונים שנקראו בפעם קודמת ואשר נקראים עכשיו.

פסוק DO לבקרת לולאות

פסוק DO מאפשר לקבץ מספר פסוקים ליחידת ביצוע אחת. את קבוצת הפסוקים אפשר לבצע כיחידה אחת פעם אחת, או מספר פעמים בלולאה, בהתאם להנחיות בפסוק DO. רק לאחר השלמת הביצוע כנדרש עובר פיקוח התוכנית לפסוק שלאחר הקבוצה.
תיחום קבוצת הפסוקים נעשה ע"י הפסוק DO בראשה וע"י הפסוק END בסופה.

המבנה הכללי של קבוצת פסוקי DO, הכוללת שני פסוקים הוא:

```
DØ --------;
     (פסוק 1)
     (פסוק 2)
END;
```

ביצוע ללא חזרה

כאשר פסוק DO אינו מכיל כל פרמטרים פרט להוראה DO עצמה, תבוצע קבוצת הפסוקים פעם אחת בלבד. השימוש בשיטה זו נפוץ בפסוקי ההתנייה IF, כאשר כתוצאה מהערכת הביטוי יש לבצע יותר מפסוק אחד בזרוע THEN או בזרוע ELSE של ההוראה.
לדוגמה:

```
IF A=20 THEN DØ;
              C = A * 2 + Y;
              PUT DATA (C);
              END;
         ELSE ----;
```

התנייה לפי ביטוי השואה

בהוראה מסוג זה מותנה ביצוע הלולאה בהערכת ביטוי השואה הנתון בפסוק DO.

```
DØ WHILE (ביטוי השואה);
            פסוק 1;
            -------
            פסוק n;
END;
```

שמוש זה בהוראת ה-DO גורם שקבוצת הפסוקים 1 עד n תחזור ותתבצע כל עוד ערך ביטוי ההשואה הוא "נכון". כאשר ערך הביטוי יהיה "לא-נכון" הפיקוח מועבר לפסוק הבא אחרי פסוק END. כל חזרה על קבוצת הפסוקים 1 עד n נקראת לולאה (1øøp).

דוגמה

```
DØ WHILE (CØDE = 5);                   1
    GET DATA (CØDE, DATE, NUM);        2
    SUM = SUM + NUM;                   3
    PUT DATA (DATE, NUM);              4
END;                                   5
```

קבוצת הפסוקים 2, 3, 4 תתבצע כל עוד ערכו של CODE יהיה שווה ל-5. כאשר ערכו יהיה שונה מ-5 יועבר רצף פיקוח התוכנית אל הפסוק הבא לאחר פסוק 5.

יש לזכור שהתנאי נבדק לפני ביצוע הלולאה, ומכאן שהוא נבדק גם לפני הביצוע הראשון שלה. ולכן, אם ערכו של CODE שונה מ-5 לפני ביצוע פסוק 1, לא יתבצעו כלל פסוקים 2, 3, 4.

דוגמה

נתונים שני ערכים A ו-B, יש להוסיף להם את הגורם 0.1 ו-0.5 בהתאמה, עד אשר מכפלתם תהייה גדולה או שוה ל-35 (או, להוסיף כל עוד המכפלה קטנה מ-35).

לשם הדגמה נציג את התכנות בשיטת פסוק ההתנייה IF ובשיטת פסוק DO.

א. פסוק IF:

```
ADD:  A = A + .01;
      B = B + .05;
      IF A * B < 35 THEN
                 GØ TØ ADD;
      PUT DATA (A,B);
NEXT: ------
```

ב. פסוק DO:

```
ADD:  DØ WHILE (A*B < 35);
         A = A + .01;
         B = B + .05;
      END;
      PUT DATA (A,B);
NEXT: ------
```

ביטוי ההשואה בפסוק DO מורה שהלולאה (או קבוצת הפסוקים) תתבצע מספר פע-מים כל עוד המכפלה A*B תהייה קטנה מ-35. עם סיום הלולאה יעבור הפיקוח להו-ראה PUT, ולאחריו לפסוק שתויתו NEXT.

התנייה עם משתנה פיקוח (אינדקס)

הפיקוח על ביצוע הלולאה נעשה על ידי משתנה פיקוח. משתנה הפיקוח מקבל ערך התחלתי מסויים אשר מקודם בכל ביצוע של הלולאה עד אשר הוא מגיע לערכו המקסימלי. בשלב זה מופסק התהליך והתוכנית ממשיכה בביצוע ההוראה שלאחר פסוק END.

```
DØ (משתנה)=(ביטוי-1) TØ (ביטוי-2)[BY (ביטוי-3)];
    פסוק -1
    --------
    פסוק -n
END;
```

ביטוי -1, ביטוי -2 וביטוי -3 הם ביטויים אריתמטיים ומייצגים ערכים מספריים. כאשר רצף התוכנית מגיע בפעם הראשונה להוראת DO, מקבל ה"משתנה" את ערך "בי-טוי-1", וערכו מושוה ל"ביטוי 2"; אם ערך זה קטן או שוה לערכו של "ביטוי 2" מתבצעת קבוצת הפסוקים 1 עד n. הוראת END מחזירה את הפיקוח להוראת DO, שם מתווסף ערכו של "ביטוי -3" ל"משתנה", ושוב מתבצעת השואה ל"ביטוי -2" וכך

הלאה. כאשר ערכו של "משתנה" גדול מערכו של "ביטוי-2" מועבר רצף הפיקוח לפסוק לאחר הוראת END.

<u>הערה:</u>

1) אם ערכו של ביטוי-3 שוה ל-1 - אפשר להשמיט את הפרמטר: (ביטוי-3) BY.

2) אפשר לכתוב את האיברים בפסוק בסדר שונה, למשל:

DØ (משתנה) = (ביטוי-1) BY (ביטוי-3) TØ (ביטוי-2);

<u>דוגמה</u>

```
1  DØ NTIMES = 1 TØ 25 BY 1;
2     GET DATA (T);
3     S = (T + 0.02) * 0.5;
4     PUT LIST (S);
5  END;
6  -----
```

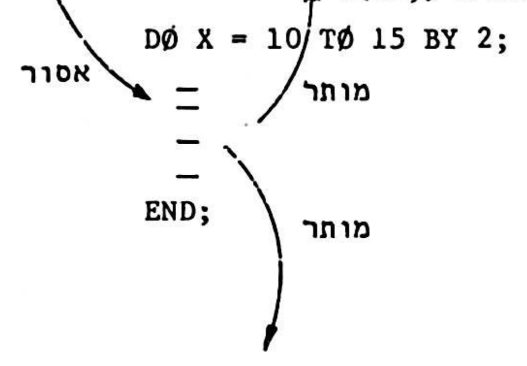
בדוגמה זו תתבצע קבוצת הפסוקים 2-4, במשך 25 פעמים. פעם הראשונה יקבל המשתנה NTIMES את הערך 1, ערכו יושוה ל-25; ובפעמים הבאות יתווסף בכל פעם הערך 1 ל-NTIMES. כאשר ערכו של NTIMES יהיה 26 לא תתבצע קבוצת הפסוקים 2-4 והפיקוח יעבור לפסוק 6.

<u>כללים לתכנות הוראת DØ</u>

א. תחום הלולאה:

מותר להסתעף מתוך תחום ה-DO אל מחוץ לתחומו. ערכו של משתנה הפיקוח נשמר וזהו ערכו <u>האחרון</u> לפני ביצוע ההסתעפות.

אסור להסתעף לתוך תחום של הוראת DO מבלי לעבור דרך הוראת ה-DO עצמה, פרט למקרה של פסוק DO ללא התניית חזרה (DO;).



ב. אפשר להשתמש במשתנה הפיקוח של הוראת DO בתוך תחום הלולאה או מחוצה לו, לדוגמה:

```
GET DATA (UPPER);
DØ X = 0 TO UPPER BY 0.25;
   PUT LIST (X, X**2)
END;
```

בדוגמה זו יודפס כל מספר בין אפס לבין ערכו של המשתנה UPPER בהפרשים של 0.25 בין מספר למספר, יחד עם הערך הרבועי שלו. קטע מפלט התוכנית יהיה (לא רשום במבנה מדוייק של הדפסת LIST):

0.25	0.0625	0.5	0.25	0.75	0.5625	1.00	1.00
1.25	1.5625	1.5	2.25	1.75	3.0625	2.00	4.00

ג. כאשר ערכו של "ביטוי 3" קטן מאפס תתבצע הלולאה אם ערכו של "משתנה" קטן מ"ביטוי 2". לדוגמה:

```
DØ Y = 25 TØ 6 BY -5;
---------
END;
```

קבוצת הפסוקים תתבצע עד אשר ערכו של Y יהיה קטן מ-6 (דהיינו Y = 5).

ד. הסתעפות מתוך תחום הלולאה אל ההוראה END מותרת, ואינה שונה מהמקרה בו מגיעים להוראה זו ברצף התוכנית. הפיקוח בלולאה יעבור לראש הלולאה (להוראת DO), ערכו של "ביטוי 3" יתווסף למשתנה הפיקוח ומחזור הלולאה המבוקר על ידי הוראת DO ימשך. בדוגמה שלפנינו יבוצע הסיעוף אל הוראת END כאשר ערכו של SN גדול או שווה ל-120 וגם SN קטן או שווה ל-130 (כלומר $120 \le SN \le 130$):

```
DØ TN = (K+5)*3 TØ 150 BY N/2
   SN = SN + TL;
   IF SN   >= 120 & SN <= 130 THEN GØ TØ SØF;
   TSN = TSN + SN*0.975;
SØF:END;
```

שים לב שבהוראת DO שבדוגמה נכללו גם ביטויים ולא רק משתנים וקבועים. המשתנ- K, N צריכים להכיל ערכים מבוקרים לפני הכניסה להוראה DO. TN יקבל בתחילה את ערך הביטוי 3*(K+5) ואח"כ, בכל מחזור של הלולאה יתווסף אליו הערך N/2 עד אשר ערכו יהיה שווה או גדול מ-150.

ה. הסתעפות מתוך תחום ה-DO לראש הלולאה, להוראה DO עצמה, זהה למצב בו מגיעים ברצף הפיקוח הרגיל להוראת DO.

```
RTRN: DØ X = 20 TØ J1 BY 3;
      -------
      IF Z < 0 THEN GØ TØ RTRN;
      ------
      END;
```

אם ערכו של Z יהיה קטן מאפס תתבצע הסתעפות ל-RTRN דהיינו, X יקבל שוב את הערך 20 (הערך ההתחלי) וכל תהליך ביצוע הוראת ה-DO יחזור על עצמו.

ו. הערכת הביטויים בהוראה נעשית פעם אחת בלבד במשך ביצוע הוראת ה-DO, וכל שינוי בערך המשתנים המופיעים בביטויים הנ"ל איננו משפיע על ערכם. ל"ביטוי-1" יש משמעות רק לביצוע הלולאה בפעם הראשונה.

דוגמה

```
     I = 8;
     N = 20;
     L = 2;
DØ J = I TØ N BY L;
     -----
     N = 100;
     ----
     L = 52;
     -----
END;
```

ערך הביטויים I, N ו-L יחושב רק בתחילת ביצוע הוראת DO וערכם ישמר במשתנים זמניים. משך כל זמן ביצוע הלולאה יהיה ערכו של "ביטוי 2" - 20 וזה של "ביטוי 3" יהיה 2, למרות שכבר בביצוע הלולאה הראשונה משתנה ערכם של N ו-L.

ז. אם משמיטים בקידוד את "ביטוי 2" ולא משמיטים את "ביטוי 3" לא תתבצע כל השואה אוטומטית כדי להפסיק את ביצוע הלולאה. במקרה זה צריך המתכנת לבצע בדי-קה כלשהי ולהתנות המשך הביצוע בבדיקה זו.

אם משמיטים גם את "ביטוי 2" וגם את "ביטוי 3" תתבצע הלולאה רק פעם אחת, לדוגמה:

```
DØ X = 25;
-----
END;
```

לולאה זו תתבצע פעם אחת בלבד כאשר ל-X יש את הערך 25.

הוראות DO משולבות

א. בהוראת DO אחת אפשר להתנות את ביצוע הלולאה בשני תנאים שונים: לפי ביטוי השוואה ולפי משתנה פיקוח. מבנה ההוראה:

```
DØ (משתנה) = (ביטוי-1) TØ (ביטוי-2) BY (ביטוי-3) WHILE (ביטוי-4);
              פסוק 1
              ------
              פסוק n
END;
```

זהו שילוב של שתי צורות הקידוד של ההוראה.

כאשר נשתמש בצורת קידוד זו תתבצע קבוצת הפסוקים 1 עד n עד אשר יתקיים אחד משני התנאים: או שערכו של ה"משתנה" יהיה גדול (או קטן) מ"ביטוי 2", או ערך "ביטוי 4" יהיה "לא נכון".

הדוגמה הבאה תמחיש אפשרות זו:

```
DØ N = 22 TØ 4 BY -2 WHILE (K  <= L);
  -------
END;
```

ערכו של N ישתנה מ-22 עד 4 בקפיצות של 2-, וערכו של K מושיה בכל חזרה לערכו של L. כאשר ערכו של N יגיע ל-2 ו/או ערכו של K יהיה קטן או שווה ל-L, יופסק ביצוע הלולאה והפיקוח יעבור לפסוק הבא אחרי הוראת END. "ביטוי 4" מוערך רק אם השואת ערכו של N ל-4 מורה על המשך ביצוע ההוראה.

ב. בהוראת DO אחת אפשר לתת תחומים שונים ותנאים שונים לביצוע קבוצת הפסוקים הכלולה בלולאה. לדוגמה:

```
DØ X = 5 TØ N WHILE (Y < 0.5), N+10, N+11, 2 TØ 7 WHILE (L=0);
  ----------
END;
```

בדוגמה זו תתבצע קבוצת הפסוקים בכמה מחזורים:

- במחזור הראשון כאשר X יקודם מ-5 עד N בקפיצות של 1 וכל עוד ערכו של Y קטן מ-0.5.
- במחזור השני כאשר X יקבל את הערך N+10 וקבוצת הפסוקים תתבצע פעם אחת.
- במחזור השלישי, כאשר X יקבל את הערך N+11 וקבוצת הפסוקים תתבצע פעם אחת.
- במחזור הרביעי כאשר X יקבל ערך התחלתי 2 ויקודם עד 7 וכל עוד ערכו של L שוה ל-0.

שיטת קידוד זו זהה בדיוק לקידוד הבא הכולל סדרה של הוראות DO, אולם הוא חוסך את הכתיבה החוזרת של ההוראות:

```
DØ X = 5 TØ N WHILE (Y < 0.5);
-------
END;
DØ X = N+10;
-------
END;
DØ X = N+11;
-------
END;
DØ X = 2 TØ 7 WHILE (L = 0);
-------
END;
```

<u>זכור</u>: השמר מפני "התחכמויות" וכתיבה מסורבלת. למרות האפשרויות הרבות לשילוב הוראות ופעולות מן הראוי לשקול את היתרונות של הכתיבה התמציתית לעומת היתרונות של בהירות ודיוק האפשריים בכתיבה פשוטה יותר. תוכנית פשוטה נוחה יותר לבפוי, ובסופו של דבר - מאפשרת תגובה מהירה יותר לדרישת ביצוע של משימת עיבוד הנתונים.

לולאות מוכלות

קבוצת הפסוקים השייכת להוראת DO אחת יכולה להכיל הוראת DO אחת או יותר. לדוגמה:

```
DØ J = 1 TØ 100 BY 1;
----------
   DØ X = 1 TØ 5;
   ---------
   END;
   ---------
   DØ I = 2 TØ 8;
      DØ K = 1 TØ 4;
      --------
      END;
   END;
----------
END;
```

הכלל לכתיבת הוראות DO במתכונת זו הוא כדלקמן:

כל הוראת END תוחמת את הוראת ה-DO <u>הקרובה אליה ביותר</u>. בדוגמה הקודמת פסוק 5 תוחם את פסוק 3, פסוק 10 תוחם את 8, פסוק 11 תוחם את 7, ופסוק 13 תוחם את פסוק 1.

מכאן ברור גם שמספר הפעמים שיבוצעו הפסוקים 1 עד 13 איננו זהה. בכל לולאה של הלולאה החיצונית (פסוק 1) יבוצעו 5 לולאות של ה-DO בפסוק 3, ו-7 לולאות של ה-DO בפסוק 7, ועל כל לולאה של זה האחרון יבוצעו 4 לולאות של ה-DO בפסוק 8, ובסה"כ יבוצע כל פסוק מספר פעמים לפי הפירוט הבא:

פסוקים 12-13, 6-7, 2-3 יבוצעו 100 פעמים.

פסוקים 4-5	יבוצעו $500 = 100 \times 5$ פעמים.
פסוקים 8-11	יבוצעו 7 פעמים.
פסוקים 9-10	יבוצעו $2800 = 100 \times 7 \times 4$ פעמים.

<u>תרגיל</u>

נתון קובץ כרטיסים המכיל ערכים שונים. הערך הראשון הוא מספר הערכים בקובץ. לדוגמה:

```
3   15.7   25   12.4
```

יש לקרוא את הנתון הראשון N, ואח"כ בזה אחר זה את N הערכים X. הנתונים מנו- קבים ברצף במבנה LIST. כל המשתנים הם מסוג FIXED DECIMAL: המשתנה N הוא בן 2 ספרות והמשתנה X הוא בן 5 ספרות מהן 2 עשרוניות. יש לחשב עבור כל X את החזקות X^2 עד X^{10} ולהדפיס אותן במבנה הבא:

ערך X^3, 3, ערך X^2, 2; ערך X = X,

כלומר, הערך של X יודפס במבנה DATA ומעריכי החזקות וערכי X בחזקה יודפסו במבנה LIST.

<u>התוכנית</u>

```
PRØBLEM: PRØCEDURE ØPTIØNS (MAIN);
    DCL N FIXED (2), X FIXED (5,2);
    GET LIST (N) /* READ CØNTRØL VARIABLE */;
READ: DØ I = 1 TØ N;
          GET LIST (X);
          PUT DATA (X);
          Y = X;
      /* NØW STARTS CALCULATIØN LØØP */
CALC: DØ L = 2 TØ 10;
          Y = Y*X;
          PUT LIST (L,Y);
       END /* CALC LØØP */;
    END/* BIG LØØP */;
END;
```

<u>הסבר</u>

1 - פסוק הפתיחה של התוכנית. שם התוכנית הוא PROBLEM.

2 - פסוק הגדרה של המשתנים N, X בהתאם לתנאי הבעיה. שני המשתנים הם דצימליים עם נקודה קבועה. אפשר להשמיט את מלת המפתח DECIMAL ולרשום רק FIXED.

3 - קריאה של המשתנה הראשון של הנתונים. N מייצג את מספר המשתנים X אשר נמצאים בקובץ. בפסוק זה כלולה גם הערה למטרת הסבר הפעולה.

4 - תחילת הלולאה הראשית של התוכנית. סיום הלולאה בהוראה END בשורה 13. את הלולאה יש לבצע N פעמים (יש לקרוא ולעבד N ערכים X מהקובץ). משתנה הבקרה של הלולאה הוא I והוא משתנה מ-1 עד N בדילוגים של 1.

5 - קריאת ערך X מתוך הקובץ (בשיטת LIST).

6 - הדפסת ערך X שקראנו בהוראה הקודמת (בשיטת DATA).

7 - העברת ערך X למשתנה Y להמשך העיבוד וכדי לא "להרוס" את ערך X אשר יהיה נחוץ לנו בהמשך.

- הערה להסבר.

9 - תחילת הלולאה לחישוב החזקות של המשתנה X. על לולאה זו יש <u>לחזור</u> עבור <u>כל</u> ערך X שמתקבל. לולאה זו מוכלת בתוך הלולאה המתחילה בשורה 4. סיומה בהוראת END בשורה 12. משתנה הבקרה של הלולאה הוא L והוא משתנה מ-2 עד 10 ועל כן הלולאה תתבצע 9 פעמים.

10 - חישוב החזקה ע"י מכפלה.

11 - הדפסת מעריך החזקה וערך החזקה עבור X בכל ביצוע של הלולאה.

END- כאשר יש מקרה, כמו שלפנינו, של הוראות END עוקבות אפשר "לחסוך" הוראות ע"י כתיבת הוראה אחת עם תוית הפסוק הפותח את הלולאה החיצונית ביותר. למשל:

```
READ: DØ -----
      --------
CALC: DØ -----
      END READ;
```

כלומר, הוראת END READ תסגור את הוראת DO הפנימית (עם תוית CALC) וגם את הוראת DO החיצונית (עם תוית READ).

בדוגמה שבתרגיל אפשר להשתמש בהוראת END של הבלוק (בשורה 14) לתחום גם את הלולאות ע"י הכתיבה END PROBLEM ולהשמיט את ההוראות END בשורות 11, 12.

פרק 5: הוראות נוספות לקלט פלט ועריכה

קלט/פלט בשיטת EDIT

בהוראות קלט/פלט מסוג DATA ו-LIST הקלט והפלט הם במבנה ובצורה מוגדרים בקלט הנתונים מופרדים זה מזה, ובפלט ההדפסה הם מורפסים בטורים קבועים בשיטת DATA נילווה גם שם-המשתנה בקלט ובפלט. בשיטת EDIT (Edit-directed I/O) אפשר להגדיר כפי הצורך את מבנה הנתונים בקלט ובפלט בכל סוג קיבץ שהוא. שיטה זו דומה לשיטת קלט ופלט הנהוגה בשפת התכנות פורטרן.

בשיטת EDIT יש למתכנת שליטה מלאה על דרך קריאת הנתונים מקובץ הקלט על עריכת הנתונים בקובץ הפלט. המבנה הכללי של הוראת EDIT היא:

לקלט - ;(רשימת מבנה) (רשימת נתונים) GET EDIT

לפלט - ;(רשימת מבנה) (רשימת נתונים) PUT EDIT

כללי הכתיבה של רשימת הנתונים (data list) זהים לכללי הכתיבה של רשימת המשתנים בהוראה מסוג LIST, ולכן לא נחזור עליהם כאן. ובקצרה. ברשימה זו אפשר לכלול משתנים, קבועים או ביטויים המותרים בשפה, לדוגמה.

A-B/3, 'KØTERET', -3, 5, MØNE, XYZ, 3*A.

לדוגמה:

GET EDIT (NAME)(A(20));

שדה NAME בכרטיס מתחיל בטור 1 והיא באורך של 20 תוים.

PUT EDIT (MØNE)(F(8,2));

שדה MONE יודפס באורך של 8 תוים הכוללים נקודה עשרונית ושתי ספרות עשרוניות

רשימת המבנה (format list) כוללת את התאור של הנתונים המופיעים ברשימת הנתונים, ובנוסף - גם הוראות פיקוח שונות. תאור הנתונים כולל את סוג הנתון (מספרי, אלפביתי) ואת אורך השדה. הוראות הפיקוח כוללות מעבר לדף חדש, שורה חדשה ומרווחים.

קלט/פלט בשיטת EDIT הינו נוח ופשוט מכיון שהגדרות המבנה סמוכות לרשימת הנתונים וכוללות אפשרויות רבות לעריכה. אולם <u>זכור</u>: שיטה זו טובה לכמות קטנה של נתונים. למטרת קלט ופלט של כמות נתונים רבה יש להעדיף שיטות עבודה אחרות - עבודה במבנים (structures) כפי שיוסבר בהמשך.

<u>רשימת נתונים ורשימת מבנה</u>

הנתונים בקובץ שנקרא ע"י הוראת EDIT או נכתב ע"י הוראת EDIT אינם מופרדים אחד מן השני, ע"י פסיק או מירווח. בהוראות EDIT נחשב <u>כל תו</u> בקלט או בפלט כחלק מן הנתונים ועל כן יש לתארו ברשימת המבנה. לדוגמה, חמישה נתוני קלט המנוקבים בכרטיס ברציפות בסדר הבא:

```
10925.101359JOHN93501011101
```

השדה הראשון הוא המספר 10925.101
השדה השני הוא המספר 359
השדה השלישי הוא המחרוזת JOHN
השדה הרביעי הוא המספר 350
שדה החמישי הוא המחרוזת 1011101

בהוראת EDIT יש צורך לתאר את סוג ומבנה הנתונים בפלט או בקלט: יש צורך לציין במפורש ש-JOHN הינו מחרוזת תוים, דבר שהיה מיותר בקלט בשיטת DATA או LIST. בשיטות אלו נהוג לרשום למשל 'JOHN' בין גרשים.

בזמן ביצוע פעולה מסוג EDIT נעשה זיווג בין רשימת הנתונים לבין רשימת המבנה (פרט למבני פיקוח) לפי מיקומם הסידורי ברשימה. לדוגמה:

```
PUT EDIT (NAME, AGE, DATE)(A(15), X(5), F(2), X(5), F(6));
```

הנתונים NAME, AGE, DATE יכתבו בקובץ הפלט לפי תאורי המבנה A(15), F(2), F(6) בהתאמה. מבנה X מיועד לפיקוח על מספר תוי הרווח בין השדות.

אם רשימת הנתונים ארוכה מרשימת המבנה, יחזור הפיקוח לתחילת רשימת המבנה עד לסיום רשימת הנתונים. לדוגמה:

```
GET EDIT(PR1, IT1, PR2, IT2)(F(5,2), A(20));
```

PR1 יקרא לפי פורמט F(5,2), IT1 יקרא לפי פורמט A(20), בנקודה זו הסתיימה רשימת המבנה ולכן יחזור הפיקוח תחילתה. הערך PR2 יקרא לפי מבנה F(5,2) ו-IT2 יקרא לפי מבנה A(20).

אם רשימת הנתונים קצרה מרשימת המבנה, דהיינו הסתיימה רשימת הנתונים ועדיין יש פריטים ברשימת המבנה, לא יבוצעו אלה האחרונים. לדוגמה:

```
PUT EDIT(T,Y,C)(F(2), X(2), F(2), X(2), A(10), X(10), F(5));
```

X הוא מבנה פיקוח לקביעת מרווח בין שדות. השדה T יכתב לפי מבנה F(2), Y יכתב

לפי מבנה F(2) ו-C יכתב לפי מבנה A(10). בנקודה זו יועבר הפיקוח על רצף התוכ-
נית להוראה הבאה, דהיינו לא תהיה שום התייחסות למבנה X(10) ולמבנה F(5).

רשימת מבנה (format list)

רשימת המבנה מכוונת לתאור מבנה הנתונים בקלט ובפלט:

א. תאור מחרוזת תוים.
ב. תאור מחרוזת סביות.
ג. תאור תמונת-תוי-עריכה (PICTURE).
ד. תאור מערכים.

בקבצי פלט להדפסה כוללית רשימות המבנה הוראות מיוחדות לעריכת תבנית ההד-
פסה ועליהן נעמוד בהמשך.

רשימת המבנה ניתנת בסוגרים, מיד לאחר רשימת המשתנים, הניתנת גם היא בסוגרים.
בין היחידות השונות שברשימה ישנם פסיקים (,) להפרדה. סדר היחידות ברשימה צריך
להתאים לסדר המשתנים שברשימה.

<u>תאור מחרוזת תוים (character-string)</u>

א. <u>תאור תוים אלפנומרים</u>

מחרוזת נתונים מתוארת ע"י התו A ובסוגרים מספר המציין את מספר התוים בשדה.
לדוגמה A(5) מציין שדה אלפנומרי באורך של 5 תוים. הגדרה זו מקבילה להגדרה
(5) CHARACTER בפסוק DECLARE. לדוגמה, יש לקרוא או לכתוב את הנתונים הבאים
במבנה תוים: 15/10/75 TEL-AVIV29
בשרשרת תוים זו שלשה שרות: תאריך, שם העיר תל-אביב ומספר בתי הקולנוע בעיר.
פסוק EDIT לקריאה או כתיבה יהיה:

```
{GET} EDIT (DATE, CITY, CINEMAS)(A(8), A(8), A(2));
{PUT}
```

ב. <u>תאור מספרים</u>

לתאור מספרים עשרוניים עם נקודה קבועה משתמשים בהגדרת המבנה F(w,d), כאשר
F מציין את אופי הנתון, w את אורך השדה בתוים, d - את מספר הספרות העשרוניות
(בימין המספר). הנקודה העשרונית יכולה להיות ממשית או לוגית

בפלט, ירשם המספר בימין השדה ואם הוא קטן מאורך השדה יתווספו מ...

רווח. אם אין ספרות עשרוניות אפשר להשמיט את הפרמטר d. אם d גדול מ-0, כלומר ישנן ספרות עשרוניות, תרשם גם הנקודה העשרונית. אם המספר מכיל פחות ספרות עשרוניות מהמקום הפנוי בשדה - יתווספו תוי אפס מימין. אם המספר שלילי ירשם הסימן מינוס (-) משמאלו. אם ישנם אפסים "מובילים" (משמאל לספרות השונות מאפס) הם יוחלפו בתוי רווח

בקלט, הנקודה העשרונית יכולה להיות רשומה או משוערת (לוגית). כאשר הנקודה העשרונית רשומה היא תקבע את ערך ומבנה המספר גם אם אלו שונים מהגדרת המבנה. אם הנקודה העשרונית אינה רשומה - מיקומה הלוגי יקבע ע"י הגדרת המבנה.

לדוגמה: F(6.2) יגדיר שדה מספרי שאורכו 6 תוים ושתי הספרות הימניות הן שבר עשרוני בפלט תתווסף תמיד הנקודה העשרונית ובקלט היא אינה חייבת להופיע. דוגמאות לשדות.

הנתון בקלט	נתון במחשב	הגדרת השדה	ערך השדה במחשב	הנתון בפלט
57.3	-	F(4,1)	57^3	-
246	-	F(4,2)	2^46	-
-	-47^3	F(5,1)	-	-47.3
-	2^9	F(4,2)	-	2.90

* שים לב שכל יחידה ברשימה מורכבת מתי המבנה ובסוגרים - מספר התוים בשדה. כל יחידה מופרדת מקודמתה בפסיק. כל הרשימה נמצאת בסוגרים. בסוף הפסוק רשום המפריד נקודה-פסיק. בדוגמה זו לא הצגנו את אפשרות הרילוג בקלט על שדות לא דרושים או רווחים בין שרות בפלט

אם אורך השדה הרשום ברשימת המבנה קטן ממספר התוים של המשתנה, יקוצצו התוים מימין. ואם אורך השדה גדול יותר - יתווספו מימין תוי רווח להשלמת המבנה המוגדר. דוגמה לקלט:

הנתון	הגדרת המבנה	תכולת השדה
AVRAHAM	A(6)	AVRAHA
1234	A(7)	1234bbb
--*	A(3)	*-*

בפסוקי פלט (PUT) אפשר להשמיט ברשימת המבנה את ציון האורך של השדה ולהסתמך על הקביעה האוטומטית של המהדיר לפי מבנה השדה בהוראת DECLARE. לדוגמה: אם ערך השדה במחשב הוא CHAR(4) וברשימת המבנה רשום התו A בלבד (ללא ציון אורך) יכתבו בפלט 4 תוים. אפשר להשתמש בשיטה זו גם לקלט: אם נתון 'ABC' אפשר להגדיר מבנה A (ללא ציון אורך) כדי לקרוא את שלשת התוים.

בהוראות קלט ופלט השונות נעשית הסבה של הנתונים, בעיקר אמורים הדברים לגבי נתונים מספריים, בין צורת הייצוג של השדה לפי רשימת המבנה לבין ההגדרה שנעשתה בפסוק DECLARE או להיפך. מספרים בשיטת הנקודה הצפה מוגדרים בצורה שונה אשר לא נעמוד עליה כאן.

*

תאור מחרוזת סביות (Bit string)

מחרוזת סביות מתוארת ע"י הגדרת המבנה B(w), כאשר w הינו מספר הסביות/תוים בשרשרת. כדי להציג מחרוזת סביות משתמשים בתוים 0 ו-1 בלבד. על כן, שרשרת בת n סביות תיוצג בקלט ובפלט ע"י מחרוזת של n תוים 0 ו-1. לדוגמה:

```
GET EDIT (CØDE)(B(5));
```

השדה בקלט מכיל 5 תוים 1,0 המייצגים סביות בהתאמה. בפלט - יוצג השדה גם כן כצרוף של תוים 1,0 בהתאמה.

תוים יקוצצו מימין או שתוי אפס יתווספו מימין כאשר השדה קצר או ארוך מהנתון המספרי.

בפלט אפשר להמנע מציון מספר התוים בשדה, והוא יהיה לפי הגדרת המשתנה בתוכנית. אם הנתון בקלט מוגדר כמחרוזת אפשר לותר על ציון האורך.

תאור תמונת תוי-עריכה (Picture)

המבנה של הגדרת תוי עריכה לשדה הינו:

P 'הגדרת התמונה'

כוונת תוי העריכה להציג את תמונת הנתונים ואת הפעולה שיש לבצע עבור כל תו ותו. למשל:

```
GET EDIT(NAME, AGE)(P'AAAAA', P'999')
```

השדה NAME מוצג ע"י 5 תוים אלפבתיים והשדה AGE ע"י 3 תוים מספריים:CHAR(n).

תוי עריכה המבצעים פעולה על שדה נתון הם V, Z, $, - ואחרים, כפי שיוסבר בסעיף עריכת נתונים, לדוגמה: PUT EDIT (CØUNT)(P'ZZZ.9');

שדה COUNT הינו מספרי ובעריכתו לפלט יוצבו תוי רווח משמאל במקום אפסים, נקוד עשרונית תוצב במקום המתאים בשדה, מימין לנקודה העשרונית תהיה ספרה עשרונית אחת. לדוגמה: שדה שערכו במחשב 0057‸83 (כמובן בלי נקודה עשרונית פיזית) יוצג בפלט כ-57.8.

תאור מערכים

פסוק קלט/פלט מסוג EDIT ושימושו עבור מערכים יודגם בפרק "מערכים".

מרווחים בהגדרת המבנה

הגדרת מרווח X(w) מיועדת לבקרת רצף התוים: בקלט - יש לדגל על מספר תוים כמפורט בהגדרה, ובפלט - מציבים תוי רווח כמספר התוים בהגדרה.

לדוגמה, אם רצף התוים בקלט הוא 927936 וההוראה היא:

GET EDIT (X,Y) (F(2) ,X(3) ,F(1)); נקבל במחשב את X=92 ואת Y=6.

אם נדפיס את X, Y לפי ההוראה

PUT EDIT (X,Y) (F(2) ,X(1) ,F(1));

נקבל את הספרות 6 92 כאשר בין ה-2 וה-6 ישנו רווח אחד.

הגדרות מקובצות

ההגדרה המקובצת מתיחסת הן לרשימת הנתונים והן לרשימת המבנה:

א. אפשר לכלול בפסוק אחד מספר רשימות:

PUT EDIT (רשימת נתונים) (רשימת מבנה) (רשימת נתונים) (רשימת מבנה) ;

לדוגמה:

PUT EDIT (ABC ,XYZ)(A(3) ,F(7))(N,ADR)(F(8,2) ,A(4));

ב. כאשר לנתונים אחדים מבנה זהה אפשר להגדיר את המבנה ע"י גורם כפל מתאים:

GET EDIT (רשימת נתונים) (n(רשימת מבנה));

לדוגמה:

GET EDIT(A,B,C,D)(3 F(7,2), A(9));

המבנה של שלשת השדות הראשונים הוא F(7,2). מבנה השדה האחרון הוא A(9).

PUT EDIT(X,Y,Z,M)(2(A(5), F(8,3)));

כאשר נדרשת חזרה על יותר מיחידת מבנה אחת, צריך לחבוק את כל הקבוצה בסוגריים. בדוגמה שלפנינו המשתנים X, Y ו-Z, M הם בעלי מבנה זהה בהתאמה, לכן חזרנו על הגדרת המבנה של X, Y <u>פעמיים</u>. שים לב לזוגות הסוגריים בהגדרת המבנה. גורם החזרה יכול להרשם <u>לפני</u> הגדרת המבנה באחד משני אופנים: כביטוי הנותן ערך מספרי, ואז יש לרשמו בסוגריים, <u>או</u> כקבוע מספרי, ואז חובה להפרידו במרווח מתו הגדרת המבנה. למשל: (5*B)F(12) או: 7 A(3).

תבנית עריכה - PICTURE

העריכה הפשוטה ביותר של נתונים נעשית ע"י הוראות קלט/פלט מסוג LIST, DATA או EDIT. הן מבצעות בקלט הסבה של שדה תוים מספרי לשדה מספרי בייצוג בי-נרי או דצימלי כמקובל במחשב, ובפלט - הסבה של שדה מספרי כפי שהוצג במחשב אל שדה תוים.

ברוב המקרים הסבה זו אינה מספיקה, ודרושה אפשרות להגדיר תבניות עריכה מורכבות יותר. תבנית עריכה של נתונים דרושה בשני מקרים:

א. בהוראות קלט/פלט מסוג EDIT directed, ע"י הציון P.
ב. בהגדרת תכונות נתונים, ע"י הציון PICTURE (או בקיצור - PIC).

עוצמתה של תכונת העריכה של הנתונים מתבטאת בעיקר בהוראות פלט להדפסה, כפי שיוסבר בהמשך. את התכונה PICTURE או P רושמים בצורה הבאה:

'הגדרה' PICTURE או 'הגדרה' P

"הגדרה" היא מחרוזת תוים הרשומה בין שני גרשים ומסמלת את אופי העריכה של הנתון. כל תו במחרוזת מקביל לתו בשדה הקלט או שדה הפלט, ולכן אורך המחרוזת זהה לאורך השדה שבו מטפלים.

מושג העריכה של הנתונים מקובל בשפת קובול, גם כן בשם PICTURE. אפשר לפנות לפרק זה לדוגמאות והסברים נוספים.

<u>הגדרות אפשריות בתבנית העריכה</u>

נכלול כאן את האפשרויות העיקריות בלבד.

א. הגדרות אפשריות לקלט ולפלט:

9 - התו המקביל בשדה יכול להיות מספרי בלבד.
A - התו המקביל בשדה יכול להיות אלפביתי בלבד.
X - התו המקביל יכול להיות מכל סוג שהוא.
V - נקודה דצימלית לוגית. כלומר היא "מוצבת" במקום היחסי בתוך השדה

דוגמאות:

```
DECLARE A PICTURE '99V9',
        B PIC     'AA9X';
```

- שדה A הוא בן 3 תוים מסוג ספרות בלבד. הספרה הימנית הינה שבר עשרוני.
- שדה B מכיל 4 תוים מהם: שנים שמאליים-אלפבתיים, אח"כ ספרה אחת והתו הימני ביותר יכול להיות מכל סוג שהוא.

ב. הגדרות אפשריות לפלט בלבד, ומכוונות בעיקר למטרת פלט הדפסה:

B יש להציב תו מרווח (blank) במקום המקביל בשדה.
S תו זה מציין שיופיע הסימן + או הסימן - משמאל לשדה, בהתאם לערכו.
+ רק אם ערך השדה הוא חיובי או אפס יופיע הסימן + משמאל למספר.
- רק אם ערך השדה הוא שלילי יופיע הסימן - משמאל למספר.
Z אם משמאל של שדה מספרי ישנם אפסים מובילים הם יוחלפו במרווחים.
* תו זה פועל כמו תו Z אולם במקרה זה יוחלפו תוי אפס בתוי *.
. נקודה עשרונית תוצב במקום המקביל בשדה.
, פסיק יוכנס למקום המקביל בשדה.
/ לוכסן יוכנס למקום המקביל בשדה.

<u>דוגמאות עריכה:</u>

השדה המקורי	תבנית העריכה	השדה הערוך	הערות לעריכה
753	999.9	75.3	השדה המקורי הותאם לתבנית העריכה.
12345	9999V.9	23450	בשדה המקורי ישנן 5 ספרות שלימים ללא ספרות עשרוניות. בתבנית נדרשה ספרה עשרונית ורק 4 ספרות שלימים.
123	+999	+123	
038	++99	+38	רק סימן + אחד נשמר בסמוך למספר.
-12	+999	12	נדרשה הדפסת סימן +
00425	***99	**425	אפסים הוחלפו בכוכביות
-001	-99	-1	
041275	99/99/99	04/12/75	תבנית עריכה זו מתאימה לתאריך.
1234	9,999	1,234	
4895	99BB99	48 95	2 מרווחים קיימים בין הספרות 8 ו-9.
9876	S99	+76	בתבנית הוגדרו רק 2 ספרות.
2468	999V99	46800	במקור לא נכללו ספרות עשרוניות ולכן הוכנסו אפסים. בתבנית צויינו רק 3 ספרות שלימים ולכן נשמטרה הספרה 2 (השמאלית) משדה המקור.

השדה המקורי	תבנית העריכה	השדה הערוך	הערות לעריכה
12345	ZZZZZ	12345	תוי Z מציינים החלפת אפסים במרווחים, והם באים במקומן של ספרות 9 בתבנית.
00834	ZZZZZ	834	אפסים משמאל הוחלפו במרווחים.
0000	ZZZ.Z	מרווחים	כל השדה הוחלף במרווחים כולל הנקודה העשרונית.
0000	ZZ9.9	0.0	רק תוים שמקבילים להם תוי עריכה Z הוחלפו במרווחים.
123.4	ZV.999	3.400	בשדה המקורי יש רק ספרה עשרונית אחת. ספרות השלימים שלא נכנסו לתבנית - הושמטו.

גורם חזרה

במקרים בהם ישנה בתבנית העריכה סדרה של תוי עריכה זהים אפשר לציין תו אחד בלבד וגורם-חזרה. גורם החזרה הינו מספר שלם הרשום בסוגריים לפני תו העריכה גורם החזרה ותוי העריכה נמצאים בין גרשיים כנדרש בתבנית. לדוגמה:

P'AAA99' זהה ל- P'(3)A(2)9'

PICTURE '*****9.9' זהה ל- PICTURE '(5)*9.9'

הגדרת תכונות שדה נתונים

שדה עריכה PICTURE מאפשר להגדיר תכונות של שדה נתונים. למשל, כאשר בקלט ישנו רצף של תוים ויש לקלוט אותו, ובאותה עת לבדוק אם השדה אלפביתי או נומרי אפשר לתכנת בצורה הבאה:

נתון:	XY384ABCD	1234AB
תבנית העריכה:	P'(4)X9(4)A'	P'(4)9(2)A'
המשמעות:	4 תוים שונים	
	1 ספרה	4 ספרות
	4 אותיות	2 אותיות

ערך של שדה מספרי אפשר להציג בשני אופנים, על ידי התכונה FIXED DECIMAL, וע"י התכונה PICTURE. בשני המקרים הצגת המספר במחשב שונה: במקרה הראשון ההצגה היא דצימלי-מקובץ (packed decimal), ובמקרה השני - בצורה של תוים בודדים בצורה מתאימה לפלט הדפסה, למשל. ערך המשתנה בשני המקרים הבאים יהיה זהה:

```
DCL  A  FIXED   DECIMAL (5,2);
DCL  B  PICTURE  '999V99';
```

השלמות לקלט/פלט בסיסי

בפרקים קודמים הצגנו שיטות קלט/פלט מסוג STREAM: DATA, LIST, EDIT. השתמשנו בצורה הפשוטה והכללית של ההוראה. הנחנו שהקלט והפלט בשיטות אלו מתבצע מ- ואל יחידות קלט/פלט סטנדרטיות מוגדרות של מערכת ההפעלה (בדרך כלל קורא הכרטיסים והמדפסת), ובפלט המכוון למדפסת לא חרגנו מהאפשרויות הסטנדרטיות של עריכת הנתונים.

אפשר לכלול בהוראות אלו פרמטרים נוספים לשם הרחבת אפשרויות העיבוד: הגדרת קובץ קלט וקובץ פלט השונים מקבצי המערכת, או דילוג לרשומה חדשה. בקובץ המיועד להדפסה יש אפשרויות נוספות לדילוג לדף חדש, לשורה רצויה, או לטור רצוי בשורה. הצורה הכללית של ההוראות היא:

<u>להוראות קלט:</u>

```
GET {DATA
     LIST
     EDIT}  FILE (שם קובץ)
            (רשימת נתונים)
            COPY - (שם קובץ)
            SKIP - (ביטוי);
```

אפשר לרשום את האופציות הרצויות בכל סדר שהוא.

לדוגמה:

```
GET DATA FILE (INVENTØRY) SKIP (5);
```

<u>להוראות פלט:</u>

```
GET {DATA
     LIST
     EDIT}  FILE (שם קובץ)
            (רשימת נתונים)
        SKIP      (ביטוי);
```

בהוראת הפלט PUT המתייחסת לקובץ <u>הדפסה</u> אפשר לציין חלק מהאופציות הבאות, או את כולן:

```
PUT {DATA
     LIST
     EDIT}  FILE (שם קובץ)
            (רשימת נתונים)
        PAGE
        LINE      (ביטוי)
        SKIP      (ביטוי);
```

בהוראות מסוג EDIT כלולים הפרמטרים בתוך רשימת המבנה, בין הגדרות השדות.
בנוסף אפשר לכלול בהוראת EDIT גם פרמטר:(ביטוי) COLUMN. דוגמאות:

```
a)  PUT FILE (REPØRT) SKIP (2) LIST (XYZ);
b)  PUT FILE (ACCØUNTS) DATA (NAME, ADDRESS) PAGE LINE (7);
c)  PUT FILE (MASTER) EDIT ('NEW RECØRD', PAGE_NØ)
            (PAGE, LINE (2), A, X(20), F(3));
```

האופציה: (שם קובץ) CØPY

אופציה זו, היכולה להופיע רק בהוראת הקלט GET, מציינת שיש לכתוב את נתוני הקלט בדיוק במבנה בו הם כתובים בקובץ הקלט, על קובץ הפלט ששמו מצויין בהוראה. אם אין מציינים שם של קובץ פלט, מניחה תוכנית המהדיר שהקובץ הוא קובץ הפלט הסטנדרטי של מערכת ההפעלה. לדוגמה:

```
GET FILE (INCØME) DATA (X,Y) CØPY (PRINT);
```

מיד לאחר קריאת הנתונים X ו-Y מקובץ הקלט הם יכתבו, בדיוק במבנה בו נקראו, על קובץ בשם PRINT.

האופציה: (ביטוי) SKIP

האופציה SKIP מציינת טיפול בשורה (או רשומה) חדשה בקובץ. הביטוי בסוגריים מוערך ומוסב למספר שלם w. הכתיבה או הקריאה יבוצעו בשורה (או רשומה) ה-w מן השורה (רשומה) הנוכחית. אם לא כותבים "ביטוי" או אם w=0 יניח המהדיר w=1 עבור קובץ שאיננו קובץ הדפסה. לגבי קבצי הדפסה, אם ערכו של w איננו גדול מאפס, יודפסו הנתונים באותה שורה בדף ההדפסה. אופצית SKIP תתבצע לפני העברת הנתונים המצויינים ב"רשימת הנתונים", אפילו אם נרשמה אחריהם. לדוגמה:

```
PUT FILE (LIMITS) LIST (A,B,C) SKIP (4);
```

הוראה זו תגרום לכתיבת תוכן הנתונים X, Y ו-Z מתחילת השורה הרביעית לאחר השורה הנוכחית.

```
PUT SKIP (3);
```

הוראה זו תגרום לקידום הכתיבה בקובץ הסטנדרטי בשלש שורות.

האופציה: PAGE

אפשר להשתמש באופציה זו רק עבור קובץ הדפסה. היא מציינת מעבר לתחילת דף חדש בקובץ ההדפסה. אופציה זו, אם מופיעה, תתבצע לפני כל אופציה אחרת בפקודת הפלט.

האופציה: (ביטוי) LINE

אפשר להשתמש באופציה זו, רק עבור קובץ הדפסה. הביטוי בתוך הסוגריים מוערך וערכו מוסב למספר שלם - w. האופציה תגרום להכנסת שורות ריקות בהדפסה, כך שהכתיבה תבוצע בתחילת השורה ה-w מתחילת הדף. אופציה זו מתבצעת לפני כל העברת נתונים, אפילו אם נרשמה לאחר "רשימת נתונים". כאשר משתמשים ב-LINE וב-PAGE, יתבצע ה-PAGE קודם. לדוגמה:

PUT LINE (42) LIST (X,Y,Z) PAGE;

ההוראה תגרום למעבר לתחילת עמוד חדש, ומשם למעבר לשורה 42 בעמוד ואח"כ יכתב תוכן הנתונים X; Y ו-Z על קובץ הפלט הסטנדרטי.

האופציה: (ביטוי) CØLUMN

אופציה זו מופעלת בהוראה מסוג EDIT בלבד. כותבים אותה ברשימת המבנה. הביטוי מחושב וערכו מוסב לערך שלם w. ערך זה מציין את המקום היחסי מתחילת השורה שבו יש לרשום את הנתון הבא ברשימה. לדוגמה:

PUT EDIT(A,B) (F(8,2), CØLUMN (25), F(2));

הנתון A יכתב בתחילת הרשומה משמאל, ובמקום ה-25 מתחילת הרשומה יכתב הנתון B.
בקלט - יקרא הנתון B החל מהמקום ה-25 מתחילת הרשומה.
אופציה זו מאפשרת לדלג על תוים ברשומות קלט ופלט. השוני מתו המבנה X הוא בכך שכאן מדלגים אל מקום יחסי מתחילת הרשומה בעוד שעם תו X מדלגים על מספר התוים הרשום עם תו המבנה.

תרגיל:

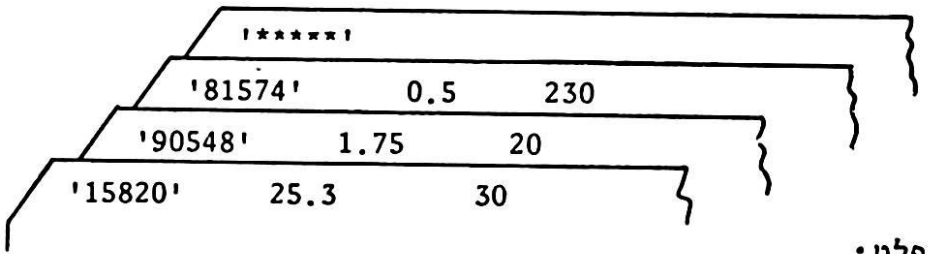
יש לכתוב תוכנית הקוראת כרטיסים, כאשר כל כרטיס מכיל שלושה שדות: א. מס' פריט (מחרוזת בת 5 תוים), ב. מחיר ליחידה (במבנה של dd.dd), ג. כמות שנמכרה (במבנה של ddd). כל כרטיס מייצג מכירה. הנתונים נוקבו מבנה המתאים לקריאה באמצעות List directed I/O.

יש לחשב את ערך המכירה (המחיר ליחידה מוכפל בכמות שנמכרה) ולהדפיס שורה עבור כל כרטיס שנקרא. השורה תכיל את השדות הבאים: א. מס' פריט, ב. מחיר ליחידה, ג. כמות שנמכרה, ד. ערך המכירה. בתחילת הדו"ח יודפסו כותרות מתאימות.

סוף הקובץ יאופיין ע"י כרטיס (אשר אין לעבדו) המכיל בשדה מס' הפריט חמש כוכביות.

בסוף הדו"ח יש להדפיס את סה"כ ערך המכירות.

דוגמה לקלט עבור התרגיל:



דוגמה לפלט:

ITEM NUMBER	UNIT PRICE	QUANTITY	VALUE
15820	25.30	30	759.00
90548	1.75	20	35.00
81574	0.50	230	115.00

TOTAL VALUE: 909.00

```
1  SALES: PRØCEDURE ØPTIØNS (MAIN);
2        DECLARE NUMBER       CHAR (5),
                 PRICE        FIXED DECIMAL (4,2),
                 QUANTITY     FIXED DECIMAL (3),
                 TØTAL        FIXED DECIMAL (9,2),
                 LØØPS_NUMBER FIXED DECIMAL (2);
3  PUT PAGE LINE (3) LIST
        ('ITEM NUMBER', 'UNIT PRICE', 'QUANTITY', 'VALUE');
4  PUT LINE (4) LIST
        ((11)'-',(10)'-',(8)'-',(5)'-');
5  TØTAL = 0;
6  DØ LØØPS_NUMBER = 1 BY 1;
7     GET LIST (NUMBER, PRICE, QUANTITY);
8     IF NUMBER = (5)'*'  THEN            /* NØ MØRE DATA */
9        DØ;
10          PUT SKIP (2) LIST
                ('TØTAL VALUE:',TØTAL);
11          GØ TØ END_ØF_PRØGRAM;
12       END;
13    ELSE;
14       DØ;
15          VALUE = PRICE*QUANTITY;
16          TØTAL = TØTAL+VALUE;
17          PUT SKIP (1) LIST
                (NUMBER,PRICE,QUANTITY,VALUE);
18       END;
19 END;
20 END_ØF_PRØGRAM: END SALES;
```

הוראות ההדפסה בפסוקים 3 ו-4 יגרמו להדפסת כותרת בראש הדו"ח. את מלת המפתח PAGE אפשר לשלב בכל הוראת פלט לקובץ הדפסה (stream I/O בלבד). משמעותה: עבור לתחילת דף חדש. סדר רישום ההוראה PAGE אינו משנה, היא תתבצע ראשונה תמיד. את ההוראה LINE(n) אפשר לשלב גם כן בכל הוראת פלט לקובץ הדפסה. משמעותה: עבור לתחילת השורה, ה-n-ית בדף ההדפסה. סדר רישום הוראה זו אינו משנה, היא תתבצע

ראשונה אלא אם כן נשתמש ב-PAGE, ואז PAGE תתבצע קודם ואח"כ LINE. בצוע פסוק 3 אם כן, יתנהל כך: בתחילה יתבצע מעבר לדף חדש, לאחר מכן יתקדם מנגנון הכתיבה לשורה השלישית בדף ואז יכתבו ארבעת הנתונים בטורים 1, 25, 49 ו-73.
אפשר היה לרשום הוראה זו כדלקמן

PUT LIST (.....) LINE (3) PAGE;

וסדר הביצוע היה זהה לזה בפסוק 3

בפסוק 4 נשתמש ב-(4)LINE כדי לעבור לתחילת השורה הרביעית בדף ההדפסה. אי שימוש ב-(4) LINE במקרה זה היה גורם לכך שהנתון הראשון ברשימה (11 סימני -) היה מודפס בהמשך שורה 3, דהיינו בטור 97.

לאחר ביצוע פסוקים 3 ו-4 יודפסו בשורה 3 ארבעה קבועי מחרוזת תוים ובשורה 4,בדיוק מתחתם,רצף סימני - באורך המתאים.

פסוקים 3, 4 ו-5 יתבצעו פעם אחת בתוכנית. פסוקים 7 עד 18 לעומת זאת, מהווים את חלק התוכנית החוזר על עצמו ותלוי במספר הכרטיסים בקובץ.

הוראת ה-DO בפסוק 6 תגרום לביצוע חוזר של פסוקים 7 עד 18 ללא תלות בתנאי כלשהו. לאחר קריאת תוכן הכרטיס הבא בפסוק 7 נבדק השדה "מס' פריט". אם תוכנו הוא '******' אין יותר נתונים (במקרה זה כרטיסים בקובץ) - נבצע את פסוקים 10, 11. בפסוק 10 מודפס ערכו הסופי של המשתנה TOTAL ולפניו, קבוע מחרוזת תוים במרחק 2 שורות מן השורה האחרונה שהודפסה. דילוג זה על שורות נעשה ע"י השימוש בהוראה SKIP(n). אפשר להשתמש בה בכל הוראת קלט/פלט מסוג stream I/O לגבי קובץ כלשהו. משמעותה: יש לדלג על n רשומות. במקרה של קובץ הדפסה זהה הרשומה לשורה.

אם יש עדיין נתונים בקובץ נבצע את פסוקים 15-17. בפסוק 15 מחושב ערכו של VALUE. בפסוק 16 מתווסף ערכו של VALUE לתוכן השדה TOTAL אשר בו אנו מסכמים את כל ערכי המכירות. בפסוק 17 מודפסים כל הנתונים הקשורים במכירה החל מתחילת השורה הבאה. השימוש ב-(1)SKIP במקרה זה מבטיח שרביעית הנתונים תודפס החל מתחילת השורה הבאה למרות שבשורה הנוכחית יש מקום (בעמודה 97) להדפסת נתון נוסף.

לפסוק 20 - סוף התוכנית - נגיע מפסוק 11, אשר יתבצע כאמור לאחר קריאת כרטיס הנתונים האחרון.

פרק 6: ארגון נתונים במערכים

ב-PL/I קיימים שני סוגי נתונים: נתונים בודדים ונתונים מקובצים בתוך מערך או מבנה. משתנה המייצג נתון בודד נקרא משתנה סקלרי (Scalar Variable) ובסוג זה של משתנים עסקנו עד כה. משתנה המייצג אוסף נתונים נקרא משתנה מערך (Array Variable) שיקרא להלן מערך, ומשתנה מבנה (Structure Variable) שיקרא להלן מבנה.

תאור המערך והגדרתו

נוח לעתים לקבץ נתונים, או משתנים בודדים, ברשימות או בטבלאות חד-ממדיות או רב-ממדיות (למשל טבלת לוח הכפל), ולהתיחס אליהם כאל קבוצה מוגדרת. כדוגמה אפשר לראות טבלה/רשימה של סוגי מכוניות האפשריות להשכרה, ובה - פריטי הרשימה. לטבלה יש שם כולל. איבריה מסודרים ברשימה סידורית וכוללים את הערכים הרצויים. כך, בשורה הראשונה ירשם שם המכונית FORD, בשורה השניה - SUSITA וכו' כמפורט בשרטוט:

CARS (1)	FØRD
(2)	SUSITA
(3)	FIAT

אפשר לרשום את ערכי הטבלה הזו בצורה הבאה:

האיבר הראשון: CARS(1) = 'FØRD'
האיבר השני : CARS(2) = 'SUSITA'
האיבר השלישי: CARS (3) ='FIAT'

אפשר לקבץ קבוצת נתונים למערך אחד אם לכולם תכונות זהות. מערך הוא אוסף חד או רב מימדי של אלמנטים או איברים שלכל אחד מהם מבנה זהה. רק המערך בכללו-תו מקבל שם, וההתייחסות לאיברים מסויימים במערך נעשית ע"י ציון מקומם היחסי במערך, כפי שמתואר בדוגמה הקודמת.

בטבלה CARS האיברים הם במבנה של מחרוזות תוים, אך אפשר לכלול בטבלה כל סוג נתונים שהוא: נתונים מספריים, נתוני מחרוזת תוים או מחרוזת סביות, תויות, מערכים.

<u>הגדרת המערך</u>

איברי הטבלה חייבים להיות רצופים כקבוצה אחת ולכן יש להגדיר במפורש את הטבלה ותכונותיה ע"י פסוק ההכרזה:

;(תכונות איברי הטבלה) (תכונות מבנה הטבלה) (שם הטבלה) DECLARE

בהוראה אחת אפשר להגדיר יותר מטבלה אחת, בדומה להגדרת משתנים.

- "שם הטבלה" יכתב לפי כללי כתיבת מזהים (משתנים),
- "תכונות מבנה הטבלה" הם ביטויים המציינים את מספר האיברים בטבלה ואת ממדיה. בטבלה רב ממדית, ירשם ביטוי לכל מימד. את תכונות המבנה רושמים <u>בסוגריים.</u>
- "תכונות איברי הטבלה", הן הגדרות תכונות של המשתנים שבטבלה. לכולם תכונות זהות.

את הטבלה CARS שהצגנו קודם נגדיר בצורה הבאה:

```
DECLARE CARS (3) CHARACTER (8);
```

דוגמה נוספת:

```
DECLARE LIST_OF_VALUES (6) FIXED DECIMAL (6,2),
        TABLE (4,2) FIXED BINARY (8);
```

בדוגמה הראשונה מוגדר מערך בשם LIST_OF_VALUES. המערך הוא בעל מימד אחד ומכיל שישה איברים. כל אחד מן האיברים הוא בעל התכונות FIXED DECIMAL (6,2). בהמשך הדוגמה מוגדר מערך שני בעל שני מימדים המכיל 8 איברים (4×2) וכל אחד מן האיברים הוא בעל התכונות FIXED BINARY (8).

המספר או המספרים העשרוניים הרשומים מיד לאחר שם המערך בתוך סוגריים מהווים את תכונת ה"מימד" (dimension attribute). תכונה זו חייבת להכתב מיד לאחר שם המערך עם או בלי מרווחים מפרידים. תכונת המימד מספקת גם את <u>מספר המימדים</u> וגם את התחום (bounds) של כל מימד.

<u>תחום</u> המימד הם ערכי תחילת המימד וסופו. <u>היקף</u> (extent) המימד הוא מספר האיברים הכולל את התחום העליון והתחום התחתון של המימד. ציון של מספר אחד בלבד בפירוט המימד במשפט ה-DECLARE, פירושו שהתחום התחתון של המימד הינו 1. בדוגמה הראשונה, המערך LIST_OF_VALUES הוא בעל מימד אחד כאמור, התחום התחתון של המימד הוא 1 והתחום העליון הוא 8, היקף המימד הוא 8. בדוגמה השניה המערך TABLE הוא בעל שני מימדים. התחום התחתון של המימד הראשון הוא 1 ותחומו העליון הוא 4. התחום התחתון של המימד השני גם הוא 1 ותחומו העליון הוא 2.

אם התחום התחתון של המימד שונה מ-1 יש לציין זאת במפורש ע"י ציון שני התחומים כשהם מופרדים ע"י המפריד (:), כבדוגמאות הבאות:

```
DECLARE VECTØR1 (-5:4) ...,
        TABLE (2:11) ...,
        MAT (-3:-1, 4, 2:6);
```

בדוגמה הראשונה VECTOR1 הוא מערך עם מימד אחד, תחומו התחתון והעליון הינם 5- ו-4 בהתאמה, והיקפו 10, וזהו גם מספר איבריו; בדוגמה השניה TABLE גם הוא מערך חד מימדי והיקפו 10, אולם תחומי המימד שלו שונים, והם 2 (תחום תחתון) ו-11 (תחום עליון). בדוגמה השלישית MAT הוא מערך תלת-מימדי; תחומי המימד הראשון הם 3- (תחתון) ו-1- (עליון) והיקפו 3. תחומי המימד השני הם 1 (תחתון) ו-4 (עליון) והיקפו 4. תחומי המימד השלישי הם 2 (תחתון) ו-6 (עליון) והיקפו 5.

מספר המימדים המירבי המותר הוא 15. הערך המזערי של תחום תחתון הוא 32768- והערך המירבי של תחום עליון הוא 32767.

התיחסות לאיברי מערך

תחומי ומימדי המערך קובעים את צורת ההתייחסות והפניה לאיברים מסויימים בו. נתייחס לדוגמה הבאה:

```
DCL X(5) FIXED DECIMAL (3);
```

נניח שברצוננו לשים בחמשת איברי X את הערכים הבאים בהתאמה: 45, 972, 5, 122, 25. פעולה זו נבצע ע"י חמישה פסוקי השמה כדלקמן:

```
X(1) = 25; X(2) = 122; X(3) = 5; X(4) = 972; X(5) = 45;
```

כל אחד מן המספרים בתוך הסוגריים המופיעים אחרי שם המערך הינו <u>ציון</u> (subscript). הציון המופיע אחרי שם המערך, עם או בלי מרווחים מפרידים, מזהה <u>איבר מסוים במערך</u>. דוגמה למערך חד-מימדי:

```
DECLARE Y(-3:2) FIXED DECIMAL (2);
```

האיבר הרביעי בטבלה זו הוא Y(0) מכיון שהאיבר הראשון הוא Y(-3). התייחסות לאיבר הרביעי, לשם הדפסת ערכו בחזקת 3, תעשה כך:

```
PUT LIST (Y(0)**3);
```

דוגמה למערך דו-מימדי:

```
DECLARE TABLE (3,4) FIXED DECIMAL (4,2);
```

מערך דו-מימדי אפשר לתאר בצורה גרפית; למימד הראשון נקרא שורות ולשני - עמודות

או טורים:

TABLE (m,n)	(m,1)	(m,2)	(m,3)	(m,4)
(1,n)				
(2,n)			TABLE (2,3)	
(3,n)		TABLE (3,2)		

התייחסות לאחד מאיברי המערך TABLE תעשה ע"י קידוד שני ציונים: ציון השורות וציון העמודות. לדוגמה: האיבר הנמצא בשורה השניה ובעמודה השלישית יצויין ע"י TABLE (2,3), וזה הנמצא בשורה השלישית ובעמודה השניה ע"י TABLE (3,2).

המחשה גרפית זו אין לה ולא כלום עם צורת ארגון איברי המערך בזכרון המחשב. המערך מאורגן בזכרון לפי סדר המימדים. כלומר, המימד השמאלי ביותר ברשימה הוא קבוע ושאר המימדים משתנים מהתחום התחתון ועד העליון. לדוגמה, במערך דו-מימדי TABLE:

TABLE (1,1), TABLE (1,2), TABLE (1,3), TABLE (1,4), TABLE (2,1), TABLE (2,2), TABLE (2,3), ..., TABLE (3,4)

תכונת "ציון" של איבר במערך היא FIXED DECIMAL (15,0). לכן, אם הוא משתנה - יש להגדירו או להשתמש בברירת המחדל למשתנים.

הציון המופיע יחד עם שם המערך אינו חייב להיות מספר קבוע, אלא <u>כל ביטוי תקף ב-PL/I</u>. כאשר ערך הביטוי אינו מספר שלם, יוסב ערכו לנתון בעל התכונות FIXED BINARY (15,0), עם השמטת החלק העשרוני של הערך המוסב

דוגמאות:

א. בדוגמה הקודמת הגדרנו את TABLE. התייחסות לאחד מאיברי TABLE יכולה להעשות ע"י

TABLE ((N+M)/X, K**2)

אם ערכם של K, X , M ,N הם 1, 5, 12, 7 בהתאמה, אזי התייחסות תהיה למעשה לאיבר TABLE (3,1). שים לב ש-N+M שוים ל-19, ותוצאת החילוק 19/5 היא 3 כאשר <u>הושמט</u> החלק העשרוני.

ב. נניח שהגדרנו מערך כדלקמן:

DECLARE MLT (20, 20, 20) FIXED DECIMAL (5);

דרוש שכל איבר במערך יכיל את ערך מכפלת שלושת הציונים המזהים אותו במערך,

מעין לוח כפל תלת-מימדי-20×20×20 קטע התוכנית שיבצע זאת יהיה:

```
1 LØØP: DØ K = 1 TØ 20;
2       DØ J = 1 TØ 20;
3       DØ L = 1 TØ 20;
4       MLT (K,J,L) = K*J*L;
5 END LØØP;
```

בדוגמה זו מופעלות הוראות DO מוכלות, כל אחת מהן פועלת על אחד מהמימדים של הטבלה MLT. ההוראה בשורה 1 פועלת על המימד הראשון, ההוראה בשורה 2 - על המימד השני וההוראה בשורה 3 - על המימד השלישי. סדר הפעולה יהיה כדלקמן:

כאשר K = 1, J = 1 ישתנה L מ-1 עד 20.
אח"כ K = 1, J = 2 ושוב ישתנה L מ-1 עד 20, וכך הלאה עד אשר
K = 1, J =20 ושוב ישתנה L מ-1 עד 20.
אח"כ K = 2, J = 1 ו-L ישתנה מ-1 עד 20. <u>הצרוף האחרון</u>
יהיה K =20, J =20, L = 20 והפעולה תפסק.

שים לב שבדוגמה זו ישנה הוראה END אחת, אשר נושאת את התוית של הוראת DO החיצונית ביותר.

ג. נניח שהוגדרו שלושה מערכים כדלקמן:

```
DECLARE (A(5), B(10), C(10)) FIXED DECIMAL (6);
```

בשלב מסוים בתוכנית יש צורך להעביר אל A את מכפלת כל איבר זוגי של B באיבר הזוגי המתאים לו ב-C. דהיינו, את B(2)*C(2) יש להעביר אל A(1), B(4)*C(4) אל A(2)- וכך הלאה. קטע תוכנית שיבצע זאת, יהיה:

```
DØ M1 = 1 TØ 5;
  A(M1) = B(M1*2)*C(M1*2);
END;
```

M1 ישתנה מ-1 עד 5 לפי אינדקס איברי המערך A. האינדקס של מערכים B ו-C ישתנה מ-2 עד 10 בדילוגים של 2 לפי המכפלה של M1*2.

הוראות קלט/פלט בסיסיות למערכים

<u>א. הוראות מסוג - Data Directed</u>

1) מבנה ההוראה לקלט:

```
GET FILE (שם קובץ) DATA (רשימת נתונים);
```

ברשימת הנתונים שבהוראה אפשר לציין את שם המערך שלתוכו מתכוונים לקרוא את הנתונים מהקובץ. שם המערך ירשם ללא כל ציון. בקובץ הקלט ירשמו הנתונים בצרוף שם המערך וציון האיברים. לדוגמה, נתונה הגדרת טבלה דו-מימדית A:

```
DCL A(5,8) FIXED (4);
```

נניח שבתוכנית יש לקרוא נתונים לתוך האיברים הבאים של A:

```
A(2,4), A(3,5), A(1,1), A(5,3)
```

הוראת הקלט לקריאת הנתונים מתוך קובץ הקלט הסטנדרטי תהיה:

```
GET DATA (A);
```

הנתונים בקובץ הקלט ירשמו בצורה הבאה, בכל סדר רצוי (אך למען הבהירות רצוי שיהיו מסודרים בצורה הגיונית כלשהי):

```
A(2,4) = 7, A(3,5) = -12, A(1,1) = 666, A(5,8) = -55;
```

2) מבנה ההוראה לפלט:

PUT FILE (שם קובץ) DATA (רשימת משתנים);

ברשימת הנתונים שבהוראה אפשר לרשום את שם המערך או איברים בודדים מתוכו לפי הדוגמאות הבאות:

ההוראה `PUT FILE (ØUTPUT) DATA (A);`

הגרום את כתיבת <u>כל</u> איברי המערך A בקובץ OUTPUT לפי הסדר בו הם מאורגנים בזכרון, דהיינו:

```
A(1,1)=666, A(1,2)=77,..., A(2,1)=-3, A(2,2)=15,..., A(5,8)=-55;
```

ההוראה `PUT FILE (RESULT) DATA (A(1,5), A(4,3));`

וגרום לכתיבת שני האיברים שברשימת הנתונים.

נתון קטע התוכנית

```
DØ ØUT  = 1 TØ 8;
   PUT DATA (A(2, ØUT ));
END;
```

קטע תוכנית זה יגרום לכתיבת <u>כל</u> איברי השירה 2 של המערך A. בצורה דומה אפשר לכתוב <u>חלק</u> מאיברי שורה, ו<u>כל</u> או חלק מאיברי טור כלשהו.

<u>ג. הוראות מסוג - List Directed</u>

1) מבנה ההוראה לקלט:

GET FILE(שם קובץ) LIST (רשימת נתונים);

ברשימת הנתונים שבהוראה אפשר לקודד את שם המערך, או שמות איברים מסוימים מתוכו.

כאשר רושמים רק את שם המערך יקראו הנתונים <u>לכל</u> איבריו. לדוגמה:

```
DCL B(8,10) CHAR (2), X(10) CHAR (7);
GET FILE (IN) LIST (B);
```

ההוראה

תגרום לקריאת 8×10 = 80 נתונים מקובץ IN והשמתם באיברי B לפי הסדר בו הם מאורגנים בזכרון.

נניח שהנתונים בקובץ אינם מאורגנים בסדר בו מאורגנים איברי B בזכרון (שורה אחר שורה), אלא בסדר הפוך, דהיינו עמודה אחר עמודה. במקרה זה אי אפשר לרשום רק את שם המערך אלא יש לפקח על סדר קריאת הנתונים.

דרך אחת לעשות זאת תהיה:

```
GET FILE (IN) LIST (((B(J,K) DØ J = 1 TØ 8)
                   DØ K = 1 TØ 10));
```

שים לב שב"רשימת הנתונים" של הוראת קלט/פלט בשיטת LIST אפשר לרשום ביטויים והוראות. במקרה זה, רשומות שתי הוראות DO: הוראת DO K היא החיצונית והו-ראה DO J מוכלת בתוכה. כלומר איברי השורות J ישתנו מהר יותר מאיברי העמו-דות K.

2) מבנה ההוראה לפלט:

PUT FILE (שם קובץ) LIST (רשימת נתונים);

ברשימת הנתונים שבהוראה אפשר לקודד גם את שם המערך ו/או שמות איברים מסויימים מתוכו, וכמובן ביטויים או קבועים.

```
PUT LIST (B);
```

ההוראה

תגרום לכתיבת כל איברי מערך B (שורה אחר שורה) בקובץ הפלט הסטנדרטי של מערכת ההפעלה. ההוראה -

```
PUT FILE (VARS) LIST ((X(N), (B(M,N) DØ M = 1 TØ 8)
                     DØ N = 1 TØ 10));
```

תגרום לכתיבת תוכן איברי X ו-B בסדר כדלקמן: האיבר הראשון של X ואח"כ כל אי-ברי העמודה הראשונה של B, לאחר מכן האיבר השני של X ואיברי העמודה השניה של B וכך הלאה.

שים לב שההוראה DO N הינה החיצונית וההוראה DO M היא ההוראה המוכלת בתוכה. לפיכך כאשר N = 1 תבוצע הכתיבה של X(1) ושל כל 8 האיברים (לפי DO M) אשר במערך B. אח"כ יהיה N = 2 ותבוצע הכתיבה של X(2) ושל כל איברי שורה 2 של המערך B וכך הלאה עד אשר N = 10.

ג. הוראות מסוג - Edit Directed

הוראות מסוג זה דומות באפשרויותיהן להוראות מסוג List.
רשימת הנתונים בהוראה EDIT נכתבת בצורה דומה לכתיבתה בהוראת LIST. באשר לרשימת תבנית הנתונים (format), נשמר הכלל שכל תבנית ברשימה זו צריכה להקביל לנתון הנמצא ברשימת הנתונים.

כאשר רשימת הנתונים ארוכה מרשימת התבנית, תחזור רשימת התבנית על עצמה עד להשלמת רשימת הנתונים. אם רשימת הנתונים קצרה מרשימת התבנית, תסתיים פעולת קלט/פלט כאשר תסתיים רשימת הנתונים.

אם נתונה טבלה TABLE בת 5 איברים, יש לרשום את ההוראה הבאה כדי לקרוא לתוכה 3 איברים:

```
GET EDIT (TABLE (I) DØ I = 1 TØ 3) ( 3  F(4,1));
```

תרגיל

התרגיל הבא ממחיש את השימוש והטיפול במערך חד-מימדי. נסוח הבעיה:
יש לקרוא 250 נתונים מספריים, כל אחד בגודל מירבי של 7 ספרות עשרוניות, לתוך מערך חד-מימדי. הקלט במבנה של List-directed I/O. יש לסרוק את איברי המערך, למצוא ולהדפיס במבנה של Data-directed I/O את הערכים הבאים: המספר המיזערי במערך ואת המספר המירבי במערך.

פתרון:

```
SEARCH: PRØC ØPTIØNS (MAIN);                                  1

        DCL  NUMBERS (250) FIXED DECIMAL (7),                 2
             (MIN,MAX) FIXED DECIMAL (7),
             INDEX FIXED DECIMAL (3);

        GET LIST (NUMBERS);                                   3
        MIN, MAX = NUMBERS (1);                               4
        DØ INDEX = 2 TØ 250;                                  5
           IF NUMBERS (INDEX) < MIN THEN                      6
                MIN = NUMBERS (INDEX);
           IF NUMBERS (INDEX) > MAX THEN                      7
                MAX = NUMBERS (INDEX);
        END;                                                  8
        PUT DATA (MIN,MAX);                                   9
END SEARCH;                                                   10
```

הסבר:

בפסוק 2 מוגדרים כל המשתנים הנחוצים לביצוע התוכנית. הראשון, NUMBERS, הוא מערך חד-מימדי בעל 250 איברים, אשר יכיל את סדרת המספרים. המשתנה השני, MIN, יכיל בסוף ביצוע התוכנית המספר המזערי בסדרה. MAX יכיל בסוף ביצוע התוכנית את המספר המירבי בסדרה. INDEX ישמש כמשתנה פיקוח בפסוק 5.

ביצוע פסוק מס' 3 יגרום לקריאה של 250 מספרים מקובץ הקלט הסטנדרטי אל 250 איברי המערך.

לאחר שהמספרים מצויים כולם בזכרון המחשב (במערך) אפשר לסרוק איבר אחר איבר במערך, ולהחליט מי ביניהם הוא המספר המיזערי ומיהו המספר המירבי. ההחלטה נעשית על סמך השואה. כיון שאי אפשר להשוות את כל המספרים במקביל, נקבע באופן שרירותי שערכו של איבר מס' 1 במערך NUMBERS הוא בעל הערך המיזערי והמירבי ונעביר לכן את ערכו אל MIN ואל MAX בהתאם (פסוק 4).

הדבר הבא שיש לעשותו הוא סריקת כל אחד מאיברי המערך והשואתו אל MIN ואל MAX. אם ערכו של איבר מסוים קטן מערכו הנוכחי של MIN, יקבל המשתנה MIN את ערכו של אותו איבר (פסוק 6). אם ערכו של איבר מסויים גדול מערכו הנוכחי של MAX, יקבל המשתנה MAX את ערכו של אותו איבר (פסוק 7).

סריקת כל איברי המערך (החל מן האיבר השני) נעשה ע"י הוראת DO (פסק 5 עד פסוק 8).

ביצוע פסוק 9 יגרום להדפסת התוצאה בקובץ הפלט הסטנדרטי. השימוש ב- Data Directed I/O מבטיח, שגם שם המשתנה יודפס, ע"י כך נוכל לזהות איזה משני המספרים שיודפסו הינו המספר המיזערי ואיזהו המספר המירבי.

ביטויים המכילים מערכים

"ביטוי המכיל מערכים" הינו ביטוי המכיל לפחות מערך אחד ותוצאת הערכת הביטוי - גם היא מערך. מערך יכול להשתתף בכל ביטוי פרט לביטוי הראשון בהוראת IF. ביטויי מערך יכולים להופיע בזרועות THEN או ELSE.

דוגמאות:

א. פעולות prefix של + או -: תוצאת פעולות מן הסוג הזה הינה מערך עם אותו מספר מימדים ותחומי המימדים כמו של המערך המשתתף בפעולה.

X ו-Y מוגדרים כמערכים כדלקמן:

```
DECLARE (X,Y) (2,3) FIXED DECIMAL (5);
```

נתון המערך X	-12	10	9
	15	-8	22

לאחר ביצוע הפסוק ;Y = -X

יהיה ערכו של Y	12	-10	-9
	-15	8	-22

ב. פעולה בין מערך לבין נתון בודד (סקלר) תפיק מערך עם אותו מספר מימדים ותחומי מימדים כמו של המערך המשתתף בפעולה.

נתייחס לדוגמה הקודמת ולפסוק ;Y = X*4

ערכו של Y לאחר ביצוע הפסוק, יהיה	-48	40	36
	60	-32	88

דהיינו, כל איבר של X מוכפל ב-4.

דוגמה נוספת תתייחס לדוגמה הקודמת ולפסוק X = X*X(1,2).

ערכו של X לאחר ביצוע הפסוק יהיה	-1200	100	90
	1500	-800	2200

שים לב! רק שני האיברים הראשונים מוכפלים בערך הראשוני של X(1,2) שהוא 10 כיון שלאחר הכפלת כל איבר של X ערכו מושם שוב ב-X, יהיה ערכו של X(1,2) לאחר גמר ה"טיפול" בו - 100, ומכאן והלאה יוכפלו האיברים הנותרים ב-100.

פעולה בין מערך למערך

כאשר יותר ממערך אחד משתתף בביטוי, חייבים להתקיים התנאים הבאים:

1. לכל המערכים יהיו אותו מספר מימדים.
2. התחום העליון והתחתון של המימד המתאים בכל מערך יהיו זהים.

<u>דוגמאות</u>

```
DECLARE (P,Q,R)(2,4) FIXED DECIMAL (2);
```

ערכו של P הוא	25	7	9	10
	7	8	5	40
ערכו של Q הוא	61	3	5	8
	18	12	28	11

א. לאחר ביצוע הפסוק ;R = P + Q

יהיה ערכו של R	86	10	14	18
	25	20	33	51

סכום האיברים המתאימים ב-P ו-Q מושם באיבר המתאים ב-R.

הפסוק הבודד R = P+Q מקביל לסדרות ההוראות:

```
DØ L1 = 1 TØ 2;
DØ L2 = 1 TØ 4;
R(L1,L2) = P(L1,L2) + P(L1,L2);
END; END;
```

אבל אין הוא חוסך בשום אופן זמן ביצוע!

ב. לאחר ביצוע ;R = P*Q

יהיה ערכו של R	1525	21	45	80
	126	96	140	440

הכפלה של כל איבר ב-P באיבר המתאים לו ב-Q והשמת ערך ההכפלה באיבר המתאים ב-R.
ההערה לעיל יפה גם כאן.

יש לזכור כי כפל זה בין מערכים <u>איננו</u> בשום פנים ואופן כפל מטריצות אלגברי!

ג. נתונה ההגדרה והנתונים הבאים:

```
DCL X(2) FIXED DECIMAL (3),
    (Z,Y)(2) FIXED BINARY (8),
    L(2) BIT (1);
```

ערכו של X הוא	2	1
ערכו של Y הוא	3	6
וערכו של Z הוא	10	2

לאחר ביצוע הפסוק:

```
L = (X + Y)**2 > Z*10;
```

יהיה ערכו של L '0'B '1'B

L(1)=1 מייצג 2**(1+6) גדול מ-   2*10,

L(2)=0 מייצג 2**(2+3) איננו גדול מ-10*10.

ד. נתונה ההגדרה הבאה של מערך, ושימושו בפונקציה:

```
DECLARE (X,Y,Z) (5,10) FLØAT DECIMAL (7);
    Y = SQRT (X+Z);
```

פונקציות בנויות ב-PL/I המקבלות מערך כארגומנט, מחזירות מערך. בדוגמה האחרונה כל איבר ב-Y יכיל את השורש הריבועי של סכום שני האיברים המתאימים לו ב-X וב-Z.

בנושא פונקציות ראה דיון בהמשך.

טיפול בחלקי מערך

אפשר להתייחס לחלק ממערך ע"י שימוש באופצית cross section. השימוש באופ-ציה יובהר להלן באמצעות הדוגמאות:

א. נניח שהגדרנו מערך דו מימדי כדלקמן:

```
DECLARE AR1 (10, 15) CHARACTER (4);
```

אפשר להתייחס לחלק מן המערך, נניח לכל השורה החמישית, כתת-מערך, העומד בפני עצמו, וכל הכללים הקשורים במערכים, חלים גם עליו;

```
PUT LIST (AR1 (5,*));
```

הוראה זו תגרום להדפסת כל השורה החמישית של AR1. התו כוכבית (*) מציין: כל היקף אותו מימד בו מופיע התו.

ב. נניח שהגדרנו מערך תלת מימדי:

```
DECLARE XAL (5,4,8) FIXED DECIMAL;
```

ע"י ההתייחסות XAL(*,*,2) מתקבל מערך דו מימדי.

ההתייחסות XAL(*,*,*) הוא, כמובן, התייחסות לכל המערך.

ערך התחלתי לאיברי מערך

אפשר להקנות לאיברי המערך ערך התחלתי ע"י שימוש בתכונה INITIAL. בסוג. העוקבים את התכונה רושמים את רשימת הקבועים שאת ערכיהם רוצים להעביר לאיברי המערך.

דוגמאות:

א.

```
DCL SWS(5) LABEL INIT (L1, START, READ, CALC, ERR);
```

האיבר הראשון בטבלת התויות SWS יקבל את הערך L1, SWS(2) יהיה START וכן הלאה.

ב. הסדר בו הקבועים מושמים באיברי המערך נקבע עפ"י ארגון המערך בזכרון המחשב נתון המערך הבא:

```
DCL PRICES (3,4) FIXED DECIMAL (5,2)
    INIT (022.15, 149.00, 895.22, 099.99, 150.50);
```

למערך PRICES 12 איברים, אולם רק לחמישה מהם ניתן ערך התחלי. אלו הם חמשת האיברים הראשונים של PRICES כפי שהם מאורגנים באחסנה, דהיינו ינתנו ערכים רק לאיברים הבאים:

PRICES (1,1), PRICES (1,2), PRICES (1,3), PRICES (1,4), PRICES (2,1).

ג. באמצעות גורם החזרה אפשר לשים קבוע אחד כלשהו ביותר מאיבר אחד. גורם החזרה חייב להיות מספר עשרוני שלם בתוך סוגריים. דוגמה:

```
DCL BIG (10,100) FLØAT DECIMAL (6),
     INITIAL ((920)0, (20) ((3)5,9));
```

920 האיברים הראשונים של BIG יקבלו ערך התחלי 0. האיברים הבאים יקבלו ערך התחלי המורכב מ-20 חזרות של המחזור 5,9(3) דהיינו 5,5,5,9 - סה"כ 20×4=80 איברים.

ד. כדי לדלג על איבר או קבוצת איברים נשתמש בתו (*) כבדוגמה הבאה:

```
DCL Y(6,3) FIXED DECIMAL INIT ((6)0, (3)*, (9)0);
```

ששת האיברים הראשונים של Y יקבלו ערך התחלי 0, בשלושת הבאים לא יושם שום ערך ובתשעה הנותרים יושם הקבוע 0.

ה. כאשר רוצים להשתמש בתכונת ה- INITIAL למערך המכיל מחרוזות יש לזכור כי בקידוד קבוע מחרוזת אפשר לציין גורם חזרה ומשמעותו שונה. לדוגמה:

```
DECLARE SER(5) CHARACTER (5) VARYING INIT ((3)'Z');
```

<u>רק</u> SER (1) יקבל ערך התחלי של 'ZZZ' והאיברים הנותרים <u>לא</u> יקבלו כל ערך התחלי! כדי "להתגבר" על בעיה זו יש לקודד בצורה הבאה:

```
DECLARE SER(5) CHARACTER (7) VARYING INIT ((5)(3)'Z');
```

ואז יקבל <u>כל</u> איבר של המערך SER את הערך 'ZZZ'.

דרך נוספת וחילופית למקרה הזה תהיה:

```
DECLARE SER(5) CHARACTER (7) VARYING INIT ((5)('ZZZ'));
```

דהיינו, הקבוע נתון בסוגריים.

<u>תרגיל</u>

יש לכתוב תוכנית למיון סדרת מספרים בסדר עולה. כל מספר הוא בן 4 ספרות. גודל מקסימלי של הסדרה הוא 50 מספרים. יש לקרוא N מספרים, למיין ולהדפיס את הסדרה הממויינת. בראש סדרת המספרים למיון נתון הערך N המציין את מספר האיברים.

<u>התוכנית:</u>

```
1   SØRT: PRØCEDURE ØPTIØNS (MAIN);
2         DECLARE (A(50), TEMP) FIXED (4);
3         GET EDIT (N, (A(I) DØ I = 1 TØ N)) (F(4));
4         X:  DØ K = 1 TØ N - 1;
5         B:  DØ J = 1 TØ N - K;
6             IF A(J) > A(J+1) THEN DØ: TEMP = A(J);
7                                       A(J) = A(J+1);
8                                       A(J+1) = TEMP;
9                                  END;
10                 END; /*B*/
11            END; /*X*/
12        PUT EDIT (A(I) DØ I = 1 TØ N)(F(5);
13  END; /* PRØGRAM SØRT */
```

<u>הסבר:</u>

בשורה 2 ישנה הגדרת הנתונים: הטבלה A היא בת 50 איברים (מקסימום) והמש-תנה TEMP. המשתנים הם מסוג FIXED DECIMAL באורך 4. מכיון שהתכונות זהות, חבוקה רשימת הנתונים בסוגריים ויש רק הגדרה אחת של התכונות.

הוראת הקריאה (שורה 3) קוראת מקובץ הקלט הסטנדרטי את הנתונים לפי שיטת EDIT. רשימת הנתונים כוללת את N (המציין את מספר האיברים בסדרה) ואת איברי הטבלה. קריאת האיברים הינה בלולאה (DO I) אשר מגדירה את N האיברים של הטבלה מ-A(1) עד A(N). מבנה הנתונים בקלט הוא 4 ספרות (F(4)), התבנית רשומה פעם <u>אחת</u> בלבד והיא תחזור עד להשלמת רשימת הנתונים. שים לב לחביקת זוגות הסוגריים בהוראה. שים לב שהמשתנה N לא הוגדר, וכמוהו גם I, K , J בהמשך - כולם מקבלים את תכונות ברירת המחדל FIXED BINARY.

הלולאה הראשית מתחילה בשורה 4. האינדקס K משתנה מ-1 עד N-1 שהוא האיבר שלפני האחרון. סיום הלולאה בשורה 11.

הלולאה הפנימית מתחילה בשורה 5 ומסתיימת בשורה 10. האינדקס בהוראה DO משתנה מ-1 עד N-K. פרוש הדבר שבפעם הראשונה לביצוע הלולאה התחום העליון של השתנות האינדקס הוא N-1. בפעם השניה, כאשר K=2, יהיה הגבול העליון N-2, וכן

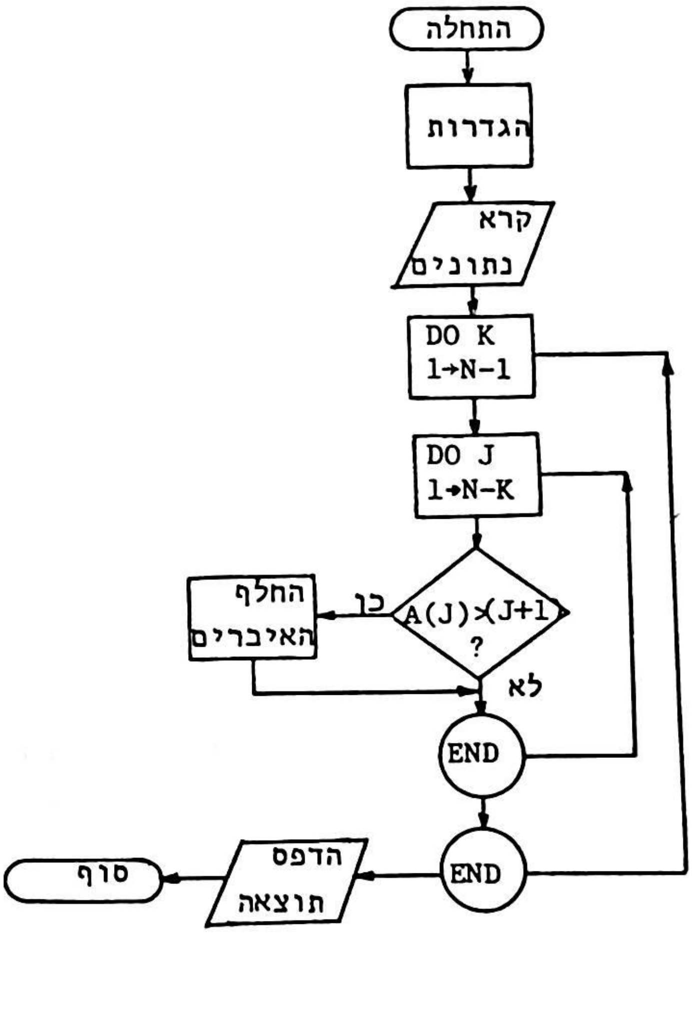
הלאה - עד אשר K=N-1 ואז הערך העליון יהיה N-(N-1)=1. בכל פעם נוספת של ביצוע הלולאה תסרק סדרת מספרים קטנה יותר מכיון שבכל פעם "דוחפים" את המספר הגבוה ביותר בסדרה אל סוף הסדרה.

השואה של זוגות מספרים והחלפת הסדר שלהם כאשר המספר הגדול, איבר J, נמצא לפני איבר J+1 — הן הפעולות העיקריות בתוכנית, שורות 6-9. אם איבר J+1 גדול מאיבר J לא נעשית כל החלפה שהיא.

בסיום ביצוע מיון המספרים לפי סדר עולה תבוצע הדפסת הסדרה בהוראה PUT בשורת 12. הדפסת הטבלה מבוצעת ע"י לולאה הרשומה ב"רשימת הנתונים". יודפסו N איברים בלבד, בהתאם למספר האיברים שנקראו. מבנה הנתונים להדפסה - 5 ספרות: כלומר 4 ספרות יוקדשו למספר עצמו ואחת תהיה מרווח. התבנית מופיעה פעם אחת והיא תחזור על עצמה עבור <u>כל</u> האיברים שברשימת הנתונים.

שים לב שהאינדקס I נוצל פעמיים בתוכנית זו, אולם ללא כל חשש של הפרעה, כי הוא סיים את תפקידו לפני שהשתמשו בו פעם נוספת.

<u>התרשים:</u>



תרגיל

נתון קובץ המכיל פרטים על פעילות מכירות של חברה במשך השנים 1971-1973. יש לערוך טבלה אשר תציג את המכירות בכל חודש בכל אחת מהשנים. בנוסף יש לסכם את המכירות לכל שנה.

תאור הנתונים:

בכרטיס ראשון מנוקבים חודשי השנה בראשי-תיבות. בכל אחד משלשת הכרטיסים הבאים מנוקבים נתוני המכירות לפי סדר השנים בכל חודש, כלומר - המכירות בחודש ינואר בשנים 1971-1973, מכירות בחודש פברואר בשנים אלו וכן הלאה.

```
JANFEBMARAPRMAYJUNJULAUGSEPØCTNØVDEC

120 234 186 199 204 213 208 197 253 244 245 196

179 208 202 --- --- --- --- --- --- --- --- ---

226 246 230 --- --- --- --- --- --- 253 258 263
```

מכיון שבכרטיס הראשון מנוקבים הנתונים ברצף, ללא רווחים אפשר יהיה לקרוא אותם בהוראת EDIT בלבד. בשלשת הכרטיסים הבאים השדות מופרדים ועל כן אפשר יהיה לקרוא בהוראת LIST.

תאור הפלט:

	1971	1972	1973
JAN	120	234	186
FEB	199	204	213
MAR	208	197	253
:	:	:	:
DEC	253	258	263
TOTALS	2580	2764	2748

IBM

PL/I Coding Form

XYZ COMPANY, SALES TOTALS
A.B. 10/1975 1 2

```
HISHUV: PROCEDURE OPTIONS (MAIN);
        DECLARE SALES (12,1971:1973) FIXED (5),
                TOTALS (1971:1973)   FIXED (5),
                MONTH (12) CHAR(3);
        PUT PAGE; /* START NEW PAGE */
        PUT EDIT (1971,1972,1973) (X(7),3 F(10));
        PUT SKIP;
        GET EDIT ( MONTH ) ( 12 A (3)); /* READ INTO TABLE */
        GET LIST (((SALES (I,K) DO K = 1971 TO 1973)
                               DO I = 1 TO 12 ));
            TOTALS = 0;
      /* LOOP TO PRINT TABLE AND COMPUTE TOTALS */
```

XYZ COMP A.B 10/1975 2 2

```
 L1:  DO I = 1 TO 12 ; /* LOOP TO PRINT EACH MONTH SALE */
            PUT EDIT (( MONTH(I) ) (SKIP, A(7))
                     (( SALES (I,K) DO K = 1971 TO 1973 ))
                       ((3) F (10) );
 L2:        DO  K = 1971,1972,1973 ;  /* ADD TO TOTALS */
                   TOTALS (K) = TOTALS (K) - SALES (I,K);
            END L2;
      END L1;
            PUT EDIT ( 'TOTALS', TOTALS )
                   ( SKIP (2), A, F(11), 2 F(10));
END HISHUV; /* END OF PROGRAM */
```

התוכנית

ראה בדף הסמוך.

הסבר

- פעולות קלט ופלט מתבצעות בקבצים הסטנדרטיים ולכן אין צורך להגדירם.
- בשורות 3, 4, 5 מוגדרים מערכים עבור הנתונים. את הנתונים נקרא למערכים אלו, נערוך בהם את החישובים ואח"כ נדפיס בהתאם להוראות התרגיל. מערך SALES הינו דו-מימדי. תחום השורות הוא 1 עד 12, ותחום העמודות הוא 1971 עד 1973. בצורה זו קבענו מימדים משמעותיים מבחינת המשתמש, לנוחיותו. מערך SALES הינו בהיקף 12×3 = 36 איברים.
- בשורה 7 ניתנה הוראה לקידום הנייר במדפסת לדף חדש.
- בשורה 9 ניתנה הוראת פלט מסוג EDIT להדפסת כותרת לעמודות. שים לב שבתחילה יש רווח של 7 תוים מתחילת השורה ואח"כ מודפסת הכותרת שבה שלשה מספרים. לאחר ההדפסה מבוצע קידום הנייר בשורה אחת כדי שיהיה רווח בין הכותרת לגוף הדו"ח (שורה 11).
- קריאת שורת שמות החודשים מבוצעת בהוראה EDIT מכיון שהם מנוקבים ברציפות ללא הפרדה. לאחר מכן מבוצעת קריאה של נתוני המכירות. אינדקס K (שורה 15) הוא בלולאה הפנימית ועל כן משתנה מהר יותר מאינדקס I אשר בלולאה החיצונית. בצורה זו תבוצע הקריאה של הנתונים כל חודש עבור כל אחת מהשנים.

בדף שני של התוכנית נמצאות הלולאות המרכזיות לביצוע החישוב וההדפסה:

- בשורות 1-7 מבוצעת הדפסת שם החודש ונתוני המכירות באותו חודש בכל אחת מהשנים. נדרשת פעולת SKIP לקידום הנייר לשורה הבאה. בהוראה אחת אפשר לכלול מספר קבו-צות נתונים, כאן הופרדה רשימת החודשים מרשימת הנתונים ולכל אחת מהן רשימת תבנית נפרדת.
- בלולאה פנימית בשורות 9-13 מבוצע סיכום של נתוני המכירות של כל חודש אל שלשת המונים השנתיים. בשורה 11 מבוקר האינדקס של איברי המערך SALES ע"י האינדקס של הלולאה הפנימית K, וע"י האינדקס של הלולאה החיצונית I.
- שורה 15 תוחמת את הלולאה החיצונית. לולאה זו תבוצע 12 פעם, לפי מספר החודשים
- עם סיום החישוב מבוצעת הדפסת התוצאות יחד עם הדפסת קבוע המציין את משמעות השורה. בתבנית ההדפסה נדרש דילוג של שתי שורות, כלומר שורת רווח ובשורה השניה תבוצע ההדפסה. שים לב שכאשר נתונה ברשימת הנתונים מחרוזת תוים בגרשיים אין צורך לציין את אורך השדה בתבנית (רשום A בבלד). כמו כן, אין צורך לציין ברשימת הנתונים של מבנה את מימדיו כאשר כותבים את כולו. ברשימת התבנית נרשם F(11) ואח"כ 2F(10) כדי לכוון את ההדפסה מתחת לעמודות (הואל נא לבדוק ולהיוכח בכך).
- סיום התוכנית בשורה 21.

פרק ז: ארגון נתונים במבנים

ב-PL/I קיימות דרכים אחדות לאחזקת אוסף נתונים, ואחת מהן הינה במשתנה מבנה (Structure Variable). צורה אחרת לאחזקת אוסף נתונים הינה במשתנה מערך (Array Variable) שנדון בפרק קודם.

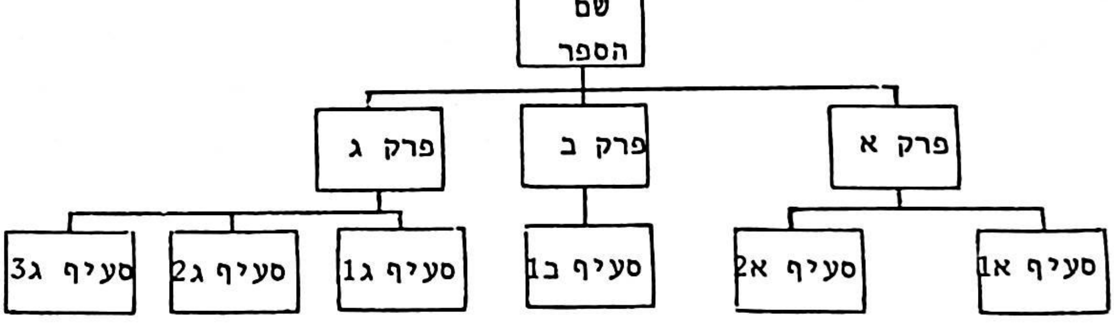
תאור המבנה והגדרתו

משתנה מערך הינו מעין טבלה, חד- או רב-מימדית, שבה כל האיברים שוים במבנה שלהם. על כן אפשר לציין שם אחד למערך כולו, ולכל אחד מאיבריו לתת ציון מספרי למקומו היחסי בתוך המערך. לדוגמה, בטבלה חד-מימדית: האיבר הראשון, השני וכו', או בטבלה דו-מימדית: האיבר שבשורה השלישית בטור החמישי וכו'.

משתנה מבנה מייצג אוסף של נתונים שיש ביניהם קשר לוגי, כפי שזה קיים במערך. אולם כאן — האיברים יכולים, ובדרך כלל הינם, בעלי תבנית (format) שונה: איברים מספריים מסוג ודיוק שונה יחד עם איברים של מחרוזות תוים וסביות. לכן כל איבר מקבל שם נפרד יחודי לעצמו וכך מזהים אותו בתוך המבנה. השימוש במבנה מקובל בעבודות מינהליות המטפלות ברשומות הכוללות נתונים מסוגים שונים.

לדוגמה, אוסף נתונים של חשבון לקוח בבנק, הכולל: מספר החשבון, שם הלקוח, כתובת הלקוח, יתרה, תאריך פעולה אחרונה. הנתונים אינם הומוגניים וכוללים ערכים מספריים ומחרוזות תוים ולכן אינם יכולים להיות מערך, אך בכל זאת מהוים יחידה לוגית אחת ששמה "חשבון לקוח" - או בשפת התכנות - מבנה (structure).

השוני העיקרי בין מערך לבין מבנה הינו בעובדה שהמבנה הינו גם מבנה היררכי של איברים, אשר כל אחד מהם יכול להיות נתון בודד או אוסף של נתונים זהים בצורה של מערך. מבחינה זו אפשר לדמות את המבנה לספר: לספר כולו יש שם אחד, הוא מחולק לפרקים אשר לכל אחד מהם שם יחודי, הפרקים מחולקים לסעיפים שגם להם שמות ייחודיים, סעיפים מסויימים עשויים להיות מחולקים הלאה לסעיפי משנה וכו'. לפנינו מבנה היררכי של ספר:



במבנה ההיראכי ישנן מספר רמות. הרמה הגבוהה ביותר הינה שם המבנה, ולכל רמה נמוכה יותר יש שם הכולל את המבנה החלקי שמתחתיה. ברמה הנמוכה ביותר בכל ענף של המבנה ישנם נתונים אלמנטריים (elementary items). הנתונים האלמנטריים מאוגדים בנתוני מבנה (structure items).

ראה גם הדיון בקובול בפרק Data Division - "תאור שדות".

הגדרת משתנה מבנה

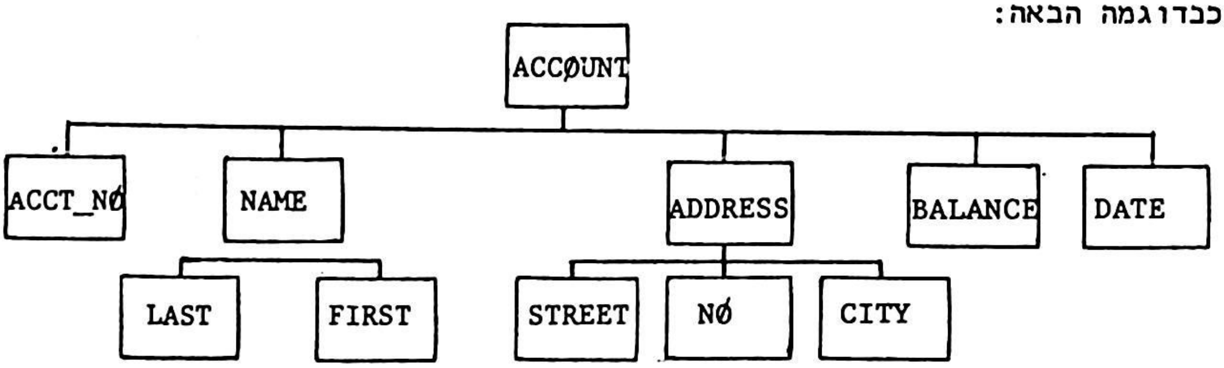
בדוגמה הקודמת הצגנו דוגמה למבנה היראכי. כעת נעבור להצגת ארגון הנתונים במבנה ולצורת הגדרתם.

כדוגמה נקח את רשומת הלקוח בחשבון עו"ש בבנק. רשומה זו כוללת את השדות הבאים:

a.	ACCØUNT_NØ	CHAR (9)
b.	LAST_NAME	CHAR (15)
c.	FIRST_NAME	CHAR (10)
d.	STREET	CHAR (20)
e.	NØ	CHAR (4)
f.	CITY	CHAR (10)
g.	BALANCE	FIXED DECIMAL (8,2)
h.	DATE	CHAR (6)

אפשר לתת לרשימת נתונים זו שם, למשל ACCOUNT ואם נכתוב את ההוראה GET DATA (ACCOUNT); נרצה שהיא תגרום לקריאת כל איברי המבנה, כלומר - כל השדות, אל זכרון המחשב.

נוח לפעמים לאגד את הנתונים/השדות בקבוצות לוגיות קטנות יותר וברמות, כבדוגמה הבאה:



ההתיחסות אל המשתנה ACCOUNT הינה אל כל איברי המבנה. פנייה אל BALANCE

הינה אל נתון זה בלבד, אך פנייה אל ADDRESS למשל, מתכוונת אל שלשת הנתונים STREET, NO, CITY.

במבנה זה ישנן רמות שונות של היראֶרכיה לשמות: שם המבנה הראשי ברמה הראשונה נקרא Major Structure Name, שמות המבנים המשניים ברמות הנמוכות יותר הם Minor Structure Names, וברמה הנמוכה ביותר בכל ענף של המבנה נמצאים שמות הנתונים האלמנטריים - Elementary Names. שמות אלו יכולים לייצג נתון רגיל (סקלרי) או מערך. רק הרמה הנמוכה ביותר מכילה נתונים (data) של ממש ולכן יוגדרו תכונותיהם. הרמות הגבוהות יותר מייצגות שמות קיבוציים לרמות הנמוכות יותר.

תיאור ארגון המבנה נעשה בפסוק ההכרזה DECLARE. ההגדרה מקבילה למבנה ההיראַרכי של שדות המבנה. מגדירים את המבנה ההיראַרכי ע"י שימוש בספרור הרמות. ספרור הרמות מתחיל במספר 1 עבור המבנה הראשי, ומספרים גדולים מ-1 ניתנים לרמות הנמוכות יותר. מספרי הרמות אינם חייבים להיות רצופים. מבנה-משני ברמה כלשהי n מכיל את כל השמות עם מספר-רמה הגדול מ-n, הנמצאים בינו לבין המבנה-המשני עם מספר רמה השווה ל-n או קטן מ-n.

מבנה הרשומה של חשבון הבנק:

ACCØUNT

ACCT_NØ	NAME		ADDRESS			BALANCE	DATE
	LAST	FIRST	STREET	NØ	CITY		
(9)	(15)	(10)	(20)	(4)	(10)	(8,2)	(6)

הגדרת המבנה בדרך הבאה:

```
DECLARE 1 ACCØUNT,
          2 ACCT_NØ CHAR (9),
          2 NAME,
            3 LAST    CHAR (15),
            3 FIRST   CHAR (10),
          2 ADDRESS,
            3 STREET  CHAR (20),
            3 NØ      CHAR (4),
            3 CITY    CHAR (10),
          2 BALANCE   FIXED DECIMAL (8,2),
          2 DATE      CHAR (6);
```

בדוגמה זו ACCØUNT הוא המבנה הראשי, NAME ו-ADDRESS הינם מבנים מש-ניים, שאר המשתנים הם אלמנטריים ורק הם תופסים מקום באחסנה הראשית ולכן רק אותם נתאר ע"י תכונות המתארות את סוג הנתון ואורכו. שים לב שלאחר כל הגדרה של משתנה יש פסיק. בסיום פסוק ההכרזה ישנו המפריד נקודה-פסיק.

תרגיל:

רשומת ציונים של תלמיד כוללת את הפרטים הבאים: מספר תלמיד, שם משפחה ושם פרטי, הכתה בה לומד התלמיד, ציונים בשלושה מקצועות. בכל מקצוע ישנם שלשה ציונים המפרטים את ציון המגן, ציון הבחינה וציון סופי.

STUDENT

STNØ	NAME		CLASS	GRADES								
	LAST	FIRST		HEBREW			BIBLE			MATH		
				1	2	3	1	2	3	1	2	3

הגדרת הרשומה תהיה:

```
DECLARE 1  STUDENT,
              3 STNØ            CHAR (6),
              3 NAME,
               (4 LAST,
                4 FIRST)        CHAR (15),
              3 CLASS           CHAR (3),
              3 GRADES,
               (4 HEBREW (3),
                4 BIBLE (3),
                4 MATH (3))
                                FIXED DECIMAL (2);
```

בדוגמה זו הוצגו מספר אפשרויות נוספות בהגדרת מבנה:

א. בשדה NAME חבקנו בסוגריים את LAST ואת FIRST מכיון שיש להם תכונת מבנה זהה - CHAR (15).

ב. בשדה GRADES חבקנו בסוגריים את שלושת המקצועות כשם שעשינו עבור שדה

NAME. במקרה זה בכל מקצוע ישנם שלשה ציונים במערך (טבלה) ולכן נרשם הציון (3), כמו למשל: HEBREW (3).

אפשר לכתוב את המשתנים בפסוק ההכרזה בשורה אחת וברצף, לדוגמה:

```
DECLARE 1 STUDENT, 2 STNØ CHAR (6), 3 NAME, (4 LAST, 4 FIRST)
        CHAR (15), ....;
```

שמות שדות במבנים שונים יכולים להיות זהים, אולם זכור, שם של מבנה חייב להיות ייחודי. כאשר שמות של שדות אינם יחודיים אין אפשרות לפנות אליהם בהו-ראות השונות שבתוכנית אלא אם מגדירים אותם ע"י שם המבנה.

לדוגמה:

```
DECLARE  1  A,
            2  B  CHAR (5),
            2  C  FIXED (7);
DECLARE  1  X,
            2  B  BIT (10),
            2  Z  CHAR (8);
```

השדה B נמצא בשני מבנים שונים ולכן נפנה אליו בצורה הבאה: שדה B של מבנה הוא A.B ושדה B של מבנה X הוא X.B.

לשדות C, Z אפשר לפנות ללא שם המבנה, לדוגמה:

```
SADE = C + 27;
```

מערך של מבנים

לעתים מורכב מערך ממשתני-מבנה ולא ממשתנים סקלריים. במקרה כזה אפשר להגדיר מערך של מבנים (Array of Structures). TABLE הינו מערך חד-מימדי של מבנה המפורט ברמות הנמוכות מ-1:

```
DECLARE  1  TABLE (5)
            (4   AAA (2)
             4   BBB),   FIXED (2);
```

ופרושו, שאוסף הנתונים לפי סדרם בזכרון הוא:

```
TABLE (1)      AAA  (1)
               AAA  (2)
               BBB
TABLE (2)      AAA  (1)
               AAA  (2)
               BBB
TABLE (3)      AAA  (1)
               ‘-----
```

מכיון ששם שדה ברמה נמוכה מופיע מספר פעמים במערך TABLE, צריך להגביל אותו ע"י הרישום של מספר ה"איבר" במערך. השדה B באיבר השני של המערך ירשם כך: TABLE(2).B, כלומר המספר הסידורי של האיבר, נקודה ושם השדה. שדה A הינו מערך ועל כן האיבר הראשון של שדה A באיבר השלישי של המערך TABLE יכתב כך: TABLE(3).A(1).

ביטויים עם מבנים

אפשר לבצע פעולות שונות עם מבנים. בעקרון, הפעולה נעשית לכל איבר בנפרד, לפי סדר האיברים במבנה.

מבנה וסקלר

הפעולה הפשוטה ביותר הינה פעולה של המבנה עם סקלר. לדוגמה, נתון המבנה:

```
DECLARE  1  A,
            2  B  FIXED (3),
            2  C  FIXED (8,2);
```

ההוראה ;A = A + 2 מתיחסת <u>לכל</u> המבנה ומבצעת את ;C = C + 2 ;B = B + 2.

ההוראה ;R = B + 8 גם כן חוקית, וכן גם ההוראה ;C = R*7.2. שתי ההוראות האחרונות מתבצעות עם המשתנים האלמנטריים של המבנה, בעוד שהדוגמה הקודמת התיחסה אל המבנה כולו.

<u>מבנה ומבנה</u>

פעולות בין שני מבנים מורכבות יותר, וכוללות למעשה שתי קבוצות פעולות: פעולות השמה ופעולות אריתמטיות.

א. פעולת השמה יכולה להתבצע בין מבנה למבנה, בין מבנה-משני אחד למבנה-משני אחר או בין משתנים אלמנטריים של המבנים.

כאשר מעבירים מבנה אחד <u>בשלמותו</u> אל מבנה אחר או מבנה-משני אחד אל מבנה-משני אחר, הם חייבים להיות זהים באיבריהם. לשמות של המשתנים ברמות הנמוכות אין כל משמעות בהוראת השמה. נתון לדוגמה:

```
DECLARE 1 A,
          2 B FIXED (6,2),
          2 C,
            (3 D, 3 E) CHAR (5);
DECLARE 1 M,
          2 N FIXED (6,2)
          2 B,
            (3 X, 3 Y) CHAR (5);
```

מכיון שהמבנים זהים בכל איבריהם: סדר האיברים, סוג הנתונים וגודלם - אפשר להעביר את כל מבנה A אל מבנה M בהוראה אחת: M = A;
המשתנים האלמנטריים B, D , E יועברו בהתאמה אל המשתנים האלמנטריים N, X, Y.
העברת שדה בודד ממבנה אחד לשני תבוצע למשל כך: X = E. ההוראה M.B = 0 (צריך לציין .M מכיון שמשתנה B נמצא גם במבנה A) תגרום לאיפוס האיברים X, Y במבנה M.

ב. פעולות אריתמטיות הקשורות במבנים אינן שונות מפעולת ההשמה, הן מתבצעות על האיברים השונים בזה אחר זה.

ג. פעולה חורגת מדרך הפעולה של טיפול בכל איבר ואיבר לפי הסדר נעשית ע"י הציון <u>BY NAME</u>

נתונים שני מבנים של פרטי תלמידים וציוניהם במקצועות שונים:

```
1 TALMID                1 STUDENT
  2 NAME                  2 NAME
  2 GRADE                 2 GRADE
    3 ENGLISH               3 MATH
    3 BIBLE                 3 ENGLISH
    3 HEBREW                3 HEBREW
    3 HISTORY               3 PHYSICS
    3 MATH
```

ההוראה: STUDENT = TALMID, BY NAME;

תגרום להעברת השדות האלמנטריים ממבנה TALMID אל השדות המקבילים ב-STUDENT, כלומר: NAME, GRADE.MATH, GRADE.ENGLISH, GRADE.HEBREW. שדות שאינם מתאימים ישארו ללא שינוי.

אפשר להתיחס גם לחלקי מבנה כמו למשל:

```
STUDENT.GRADE = TALMID.GRADE, BY NAME;
```

אפשר גם לבצע פעולות אריתמטיות עם הציון BY NAME.

ד. שימוש במבנים בהוראות קלט/פלט אינו שונה מפעולות אחרות:

```
GET LIST (TALMID);
```

יבצע את קריאת הנתונים אל תוך המבנה אשר הוגדר בדוגמה הקודמת.

אפשר גם לכתוב PUT DATA (STUDENT.GRADE);.

תכונות נוספות בהגדרות נתונים

<u>התכונה DEFINED:</u>

תכונה DEFINED (בקיצור - DEF) מציינת שמשתנה סקלר, מערך או מבנה מוגדר באותן כתובות הזכרון של משתנה אחר. לדוגמה:

```
DECLARE LIST CHAR (10),
        NAME CHAR (10) DEFINED LIST;
```

כלומר, שדה NAME ושדה LIST מתיחסים <u>לאותם</u> הנתונים, אשר אפשר לפנות אליהם בשני שמות.

אפשר גם לחפוף חלק של שדה, לדוגמה:

```
DECLARE NAME       CHAR (25),
        LAST_NAME  CHAR (15) DEFINES NAME,
        FIRST_NAME CHAR (10) DEFINED NAME PØSITIØN (16);
```

<u>התכונה LIKE:</u>

תכונה זו מציינת שהמשתנה המוגדר מקבל בדיוק את התכונות של משתנה אחר. המשתנה יכול להיות מכל סוג שהוא. לדוגמה:

```
DECLARE 1 STØCK_ITEM,
          2 CATALØG   CHAR (8),
          2 NAME,
            3 HEBREW  CHAR (30),
            3 ENGLISH CHAR (40),
          2 PRICE FIXED (8,2);
DECLARE 1 NEW_STØCK LIKE STØCK_ITEM;
```

ל-NEW_STOCK יש המבנה הבא:

```
1 NEW_STØCK,
    2 CATALØG     CHAR (8),
    2 NAME,
        3 HEBREW  CHAR (30),
        3 ENGLISH CHAR (40),
    2 PRICE  FIXED  (8,2);
```

<u>התכונה INITIAL</u>

תכונה INITIAL (או INIT) הוזכרה בהקשר למשתנים סקלריים והיא מיושמת גם במשתני מערך:

לדוגמה:

```
DECLARE NAME CHAR (12) INITIAL ('MØSHE RABENU');
```

כנתון השמה בפסוק עם תכונה INIT אפשר לכלול קבועים, ביטויים ופונקציות.

דוגמה למשתני מבנה:

```
DECLARE 1  A,
           2  B        INIT (0),
           2  C (3),
            3 D        INIT (0),
            3 E        INIT ((3)0);
DECLARE    M (15) CHAR (4) INIT ((10) 'ABCD');
```

במבנה A מוגדרים כל השדות בברירת המחדל שלהם. שדה B יאופס. מבנה-משני C הינו מערך בן 3 איברים, ולכן איפוס שדה D כפי שהוא רשום יגרום לאיפוס C.D(1) בלבד, בעוד אשר שדות E יאופסו בכל שלושת איברי המערך, כי נרשם בתכונה INIT הציון (3).

במערך M יקבלו רק 10 האיברים הראשונים את הערך ABCD.

פרק 8: קלט ופלט ברשומות

עקרונות העיבוד ברשומות

הוראות קלט ופלט בשיטת Stream כונו הוראות "קלט ופלט בסיסיות". הן טפלו בעיקר בשדות או בנתונים בודדים מתוך רצף הנתונים. אמנם אין כל מניעה לפעול עם הוראות אלו על רשומות, אך למטרה זו ישנן הוראות מתאימות ונוחות יותר. אלו הן הוראות מסוג Record-Oriented I/O. שיטת קלט ופלט זו דומה במידה רבה לשיטת קלט ופלט של רשומות בקבצים הנהוגה בקובול, והקורא מופנה אל הדיון בנושא זה.

היתרון החשוב בשיטת קלט ופלט ברשומות היא העברה ישירה ללא הסבה של נתונים מיחידת קלט לזכרון המחשב ומזכרון המחשב ליחידת הפלט. יש לזכור שבשיטות קלט/פי אחרות בוצעה הסבה של נתונים מספריים מייצוג של מחרוזת תוים לשיטת המספרים הפני-מית במחשב (דצימלי, בינרי) בהוראת קלט, ולהיפך - בהוראת פלט.

שיטת העיבוד ברשומות מקובלת בעיקר בקבצי המערכת המיועדים לעיבוד חוזר, ועל כן הם שומרים את הנתונים במבנה הדרוש לעיבוד במחשב, מכיון שממילא אינם מיועדים לטיפול של בני-אדם. ההסבה אשר נעשית מרשומות במבנה קלט אל שיטת העי-בוד במחשב וכדומה בהסבה לצורך פלט, הינה יעילה יותר ע"י פעולת השמה, מאשר ע"י הוראת קלט/פלט. כלומר, נתונים מספריים המנוקבים בכרטיס יקראו למחשב במבנה של תוים, והם יוסבו למבנה דצימלי או בינרי פנימי ע"י פסוק השמה. וכדומה — בבי-צוע פלט להדפסה, למשל.

מאגרי קלט ופלט

לצורך פעולות קלט ופלט בשיטת רשומות נדרש מאגר או שטח זכרון אשר יקלוט את רשומת הקלט או ישמש לאחסנת רשומת הפלט עד לכתיבתה. לצורך זה משמש שטח הזכרון המוגדר כמבנה ע"י הוראת ההכרזה. עבור קריאה וכתיבה ביחידות כמו קורא כרטיסים ומדפסת שיטה זו נוחה ומספיקה. אך לא כך הדבר במתקני אחסון מגנטיים מהירים.

הרשומות בקובץ הנמצא במתקן מגנטי (סרט, דיסק) מאוגדות בבלוקים. כל בלוק מכיל קבוצה של רשומות, וזאת כדי לייעל את פעולת הקלט/פלט ולחסוך בשטח אחסנה בסרט או בדיסק. כאשר מבצעים פעולת קריאה יקרא בלוק שלם אל הזכרון לשטח

שיוקצה מראש ע"י המהדיר. כל פעולת קריאה נוספת תעביר למבנה את הרשומה הבאה עד לסיום הרשומות בבלוק. אז יקרא הבלוק הבא, וכן הלאה... . בביצוע פעולת פלט, תועבר הרשומה אל שטח הבלוק המוקצה בזכרון. כאשר הוא "יתמלא" הוא יכתב בקובץ, רשומה חדשה תכתב בראש הבלוק וכן הלאה. הוראת CLOSE לקובץ פלט תכתוב את הבלוק האחרון גם אם איננו "מלא".

מקובל בדרך כלל שיהיה יותר מבלוק אחד לקלט או לפלט. והסבה - מכיון שמערכת ההפעלה יכולה לנהל פעולות חופפות היא מבצעת "קריאה" של בלוק חדש מראש בעוד התוכנית מטפלת בבלוק הקודם. אח"כ מוחלף הפיקוח, התוכנית מטפלת בבלוק שנקרא ומערכת ההפעלה "מכינה" בלוק חדש. כדומה בפעולת פלט - מערכת ההפעלה כותבת את הבלוק ליחידת הפלט בעוד התוכנית "מכינה" עבורה בלוק חדש. פעולת מערכת ההפעלה "שקופה" למתכנת.

ביצוע הוראת פתיחה לקובץ קלט, תגרום לבדיקת התוית שלו ע"י מערכת ההפעלה, ואם מתאים - תציב אותו על ראשית הנתונים לקריאת הבלוק הראשון. הוראת סגירה לקובץ פלט תבדוק את תוית הסיום של הקובץ לפי נוהלי המערכת.

ביצוע הוראת פתיחה לקובץ פלט, תרשום תוית חדשה בראשית הקובץ ותציבו בצורה מתאימה לכתיבת הבלוק הראשון. הוראת הסגירה תכתוב את הבלוק האחרון שעדיין לא הושלם, ולאחר מכן - את תוית הסיום של הקובץ.

הגדרת תכונות קובץ

בפרק קודם הוצגו כמה מהתכונות החשובות של הקובץ אשר התיחסו בעיקר לקבצים הסטנדרטיים של המערכת הפועלים בשיטת stream I/O. לקבצים ישנן מספר תכונות המאפיינות אותם הן מן הבחינה הלוגית והן מהבחינה הפיזית.

התכונות הלוגיות להגדרת קובץ הן: פלט/קלט/עדכון, שיטת stream או record, שיטת עיבוד רציפה או ישירה, שימוש במאגרים וכדי'. התכונות הפיזיות מציינות את שיטת הארגון הפיזי של הקובץ, מספר המאגרים, מבנה רשומות ובלוקים וכדי'.

ההבחנה הראשונה בין סוגי קבצים הינה בצורת הניהול שלהם, ע"י התכונה STREAM או RECORD. לכל אחת מתכונות אלו נלוות תכונות אחרות. נציין רק חלק מהן:

	RECORD I/O	STREAM I/O
סוג הפעילות	INPUT/OUTPUT/UPDATE	INPUT/OUTPUT
לקובץ הדפסה בלבד		PRINT
שיטת עיבוד	SEQUENTIAL/DIRECT	SEQUENTIAL/DIRECT
שימוש במאגרים	BUFFERED/UNBUFFERED	
ארגון פיזי	ENVIRONMENT (רשימת תכונות)	ENVIRONMENT (רשימת תכונות)

- קיצור של SEQUENTIAL הוא SEQL
- קיצור של ENVIRONMENT הוא ENV
- קיצור של BUFFERED הוא BUF

נבהיר בקצרה את רשימת התכונות:

- "סוג הפעילות" מציין את ההרשאה להפעיל על קובץ הוראות קלט/פלט מסויימות. אם ניתנה הרשאה ל-INPUT, לא תורשה פעולת PUT בקובץ זה.
- התכונה PRINT לקובץ בשיטת stream מאפשרת להפעיל או "לדכא" הוראות פלט ספצי-פיות למדפסת. כאשר מדכאים פעולות אלו (SKIP למשל) אפשר לבצע פלט אל קובץ מגנטי או כרטיסים ללא צורך בשינוי ההוראות.
- "שיטת העיבוד" מורה שהגישה לרשומות תהיה לפי סדרן או לפי מפתח זהוי. במקרה השני - נדרש ציון של המפתח בהוראות קלט/פלט.
- "שימוש במאגרים" נדון בתחילת סעיף זה. הוא מיועד בעיקר לקבצים מגנטיים בשי-טת עיבוד רציפה.
- "ארגון פיזי" גורר אחריו רשימת תכונות כפי שנפרט בהמשך.

הגדרת תכונות פיזיות לקובץ נעשית ע"י המציין ENVIRONMENT בהגדרת הקובץ. במערכות הפעלה שונות יש שוני ברשימת התכונות, מכיון שבכמה מהן אפשר להגדיר את התכונות ע"י פרמטרים הנמסרים למערכת ההפעלה בזמן ביצוע התוכנית.

הגדרת תכונות זו מספקת למהדיר את המידע הדרוש לו לשם קביעת השיטה ופקודות המכונה הנדרשות לקלט ופלט של הנתונים. הקורא ייטיב אם יפנה בנושא זה לספרות המתאימה, ובמיוחד לחוברות הנושאות את השם PL/I Programer's guide. רשימת התכונות למציין ENV הינה ארוכה. נזכיר כאן את החשובות שבהן, המאפיינות את מבנה הרשומה ואורכה ואת גודל הבלוק.

FS או F רשומות בודדות באורך קבוע.

FBS או FB רשומות באורך קבוע בבלוקים באורך קבוע.

V רשומות בודדות באורך משתנה.

VB רשומות באורך משתנה בבלוקים (באורך משתנה).

U רשומות במבנה לא מוגדר.

RECSIZE (n) אורך הרשומה בתוים. אורך קבוע או אורך מקסימלי.

BLKSIZE (n) אורך הבלוק בתוים אורך קבוע או אורך מקסימלי.

BUFFERS (n) מספר המאגרים לבלוקים של הקובץ (אפשרי יותר מ-2).

REGIONAL/VSAM/CONSECUTIVE/INDEXED שיטת הארגון הפיזי של הקובץ.

<u>ברירות מחדל בהגדרת קובץ</u>

בדרך כלל אין צורך לציין את כל רשימת התכונות ואפשר להסתפק בציון תכונות המפתח בלבד, אשר גוררות ברירות מחדל שונות. כאשר לא מציינים דבר, משתמעות התכונות הבאות:

STREAM, INPUT, SEQUENTIAL (SEQL)

- כאשר מגדירים SEQL נוספת התכונה BUFFERED
- כאשר מגדירים DIRECT נוספת התכונה UNBUFFERED

קובץ המוגדר בשיטת ניהול <u>RECORD</u>, כלומר קובץ רשומות, יקבל את רשימת התכונות הבאות כברירת מחדל:

SEQUENTIAL, BUFFERED, INPUT.

<u>דוגמאות להגדרת קבצים</u>

הגדרה מלאה לקובץ:

```
DECLARE INVENTØRY FILE SEQL RECØRD ØUTPUT BUFFERED
    ENV (FB   RECSIZE   (120)   BLKSIZE (2400) CØNSECUTIVE BUFFERS (2));
```

קובץ INVENTORY הוא קובץ במבנה רשומות שיעובד באופן רציף כקובץ פלט, עם מאגרים לבלוקים; הרשומות והבלוקים הם באורך קבוע, אורך הרשומה 120 תוים ואורך הבלוק הוא 2400 תוים, הארגון הפיזי שלו הוא רציף, ומוגדרים עבורו 2 מאגרים לבלוקים.

אפשר להשמיט חלק גדול מהתכונות ולציין:

```
DECLARE INVENTØRY FILE RECØRD ØUTPUT
    ENV (FB   RECSIZE   (120) BLKSIZE (2400));
```

הגדרה נוספת:

DCL HISTØRY FILE SEQL UPDATE ENV (.....);

את תכונות הקובץ מגדירים בפסוק ההגדרה DECLARE FILE... אולם אפשר להגדיר את תכונות הקובץ גם בפסוק הפתיחה OPEN, פרט להגדרת התכונות הפיזיות - ENVIRONMENT.

דוגמה:

ØPEN FILE (NAMES) RECØRD INPUT SEQL BUFFERED;

הוזכר שחובה, ורצוי מבחינתו של המתכנת, לבצע פתיחה של כל קובץ בתחילת העיבוד. אך אם פעולה זו לא מבוצעת וחסרות הגדרות בפסוק ההגדרה של הקובץ, יניח המהדיר שהקובץ נושא את התכונות המתאימות להוראות קלט ופלט המופעלות עליו. לדוגמה, אם נכתב GET - מניחים INPUT, STREAM; הוראות READ ו-WRITE יגררו תכונות RECORD ואת התכונות הנלוות לה.

טיפול בקובץ מוגדר

כאשר מגדירים פעולות קלט ופלט בקבצים השונים מהקבצים הסטנדרטיים של המערכת יש לבצע מספר פעולות הכנה לקראת העיבוד שלהם:

א. <u>הגדרה</u>

DECLARE (שם קובץ) FILE (תכונה);

- "שם הקובץ" נרשם לפי כללי רישום מזהים (ללא סוגריים).
- "תכונה" (בסוגריים) יכולה להיות INPUT, OUTPUT או UPDATE (תכונה המאפשרת קלט ופלט).

התכונה PRINT יכולה להחליף את התכונה OUTPUT. כאשר ישנן בהוראת הפלט פעולות המכוונות למדפסת, כגון SKIP או PAGE, הן תופעלנה רק כאשר ההוראה מכוונת לקובץ הפלט הסטנדרטי, או כאשר מוגדרת התכונה PRINT. דוגמה:

```
DECLARE PELET FILE PRINT,
        TNUA FILE INPUT;
```

ב. <u>פתיחה וסגירה</u>

כדי לבצע עיבוד קובץ השונה מהקובץ הסטנדרטי של מערכת ההפעלה יש לבצע שתי הוראות:

פתיחה - לפני ביצוע פעולה כלשהיא   ØPEN FILE (שם קובץ);

סגירה - לאחר סיום הפעולות בקובץ   CLØSE FILE (שם קובץ);

כאשר קובץ משנה את תפקידו במהלך התוכנית, וחשוב לציין זאת, אפשר לרשום את התכונה בהוראה OPEN, כבדוגמה הבאה. אם לא רושמים את ההוראה CLOSE במפורש, "נסגרים" הקבצים אוטומטית עם סיום התוכנית. "שם קובץ" ירשם בסוגריים.

תרגיל

תאור הבעיה רשום כהערה, בגוף התוכנית:

```
INØUT: PRØCEDURE ØPTIØNS (MAIN);
      /* EXAMPLE: READ 3 NUMBERS FRØM STD INP FILE AND WRITE THEM ØN
         FILE CALLED-ZMANY.  THEN READ THESE NUMBERS AND PRINT THEM */
      DECLARE (X,Y,Z,A,B,C) FIXED,
              ZMANY FILE;
      GET LIST (X,Y,Z);

      ØPEN FILE (ZMANY) ØUTPUT;
      PUT FILE  (ZMANY) LIST (X,Y,Z);
      CLØSE FILE(ZMANY);

      ØPEN FILE  (ZMANY) INPUT;
      GET  FILE  (ZMANY) LIST (A,B,C);
      PUT DATA (A,B,C) SKIP;/* SKIP TØ NEW LINE BEFORE PUT */
      CLØSE FILE (ZMANY);
END INØUT;
```

הסבר:

- הקובץ ZMANY יוגדר ליחידת ציוד המאפשרת קלט ופלט, כמו למשל סרט מגנטי או דיסק.
- קובץ ZMANY מוגדר כ-FILE בלבד. את התכונה הוא יקבל בהוראת OPEN.
- כאשר "פותחים" קובץ כ-INPUT אפשר לפנות אליו בהוראות GET בלבד, וכאשר פותחים אותו כ-OUTPUT אפשר לפנות בהוראות PUT בלבד.
- לאחר "סגירת" קובץ אי אפשר לפנות אליו בהוראות קלט/פלט.
- הוראת OPEN מציבה את הקובץ כדי שאפשר יהיה לקרוא אותו מתחילתו.

הערה:

בשלב זה כללנו רק את התכונות הבסיסיות של קבצים עבור עיבוד בשיטת stream.

סיום עיבוד קובץ קלט

ישנם ציונים מוסכמים לסיומות של קבצי קלט: בכרטיסים - זהו הכרטיס המסומן עם */ בטורים 1-2, בקבצים בסרטים ובדיסקים - תויות סיום. מערכת ההפעלה יכולה לזהות סיומת זו ולהפנות את הפיקוח, ללא התערבות המתכנת, אל קטע תוכנית מוגדר המטפל במצב של סיום הקובץ.

ההוראה לביצוע פעולת הבדיקה והפעולות הנלוות אליה נכללת במסגרת "פסוקי התניה - ON".

פסוק ההתנייה לסיום קובץ קלט יהיה:

ØN ENDFILE (שם קובץ) (פסוק/ים);

ולדוגמה:

```
ØN ENDFILE (TNUA) GØ TØ ABC;
```

או:

```
ØN ENDFILE (KØVETZ) BEGIN;
                   ----------
                   GØ TØ FINISH;
                   END;
```

את הפסוק ...ON ENDFILE אפשר לרשום בכל מקום בתוכנית. אולם בדרך כלל רצוי לרשום אותו בתחילתה, יחד עם ההגדרות האחרות. במהלך התוכנית אין כל טי-פול בנושא סיומת הקובץ והסיעוף אל קטע התוכנית המטפל בנושא זה יעשה אוטומטית ע"י הכוונת הפסוק ...ON ENDFILE.

הוראות קלט ופלט לקובץ רשומות

הוראות קלט ופלט לקובץ רשומות הן: READ, WRITE, המקבילות להוראות GET, PUT לקובץ stream.

מבנה ההוראות:

READ FILE (שם קובץ) INTØ (משתנה);

WRITE FILE (שם קובץ) FRØM (משתנה);

- "משתנה" יכול להיות משתנה סקלרי, משתנה מערך או משתנה מבנה.
- "שם קובץ" ו"משתנה" ירשמו בתוך סוגריים.

ולדוגמה, הוראות קלט/פלט:

```
READ  FILE (STUDENTS) INTØ (RESHIMA);
WRITE FILE (HISTØRY)  FRØM (RESHIMA);
```

נדגים תוכנית המבצעת פעולות קלט/פלט:

התוכנית קוראת רשומות מקובץ, וכותבת את הרשומות אל קובץ פלט. אין כל פעולות עיבוד פרט לספירת הרשומות והדפסה בסיום העיבוד.

```
1  RDWT: PRØCEDURE ØPTIØNS (MAIN);
2        DECLARE FIELD FIXED (2) INIT (0),
                 1 PARIT,
                 2 MISPAR CHAR (8),
                 2 PRICE,
                   3 IL  FIXED (8,2),
                   3 US$ FIXED (7,2);
3        DECLARE AA FILE INPUT, BB FILE ØUTPUT;
4            ØN ENDFILE (AA) GØ TØ SØF;
5        ØPEN FILE (AA) RECØRD, FILE (BB) RECØRD;
6  RD:   READ FILE (AA) INTØ (PARIT);
7            FIELD = FIELD + 1; /* CØUNT NØ ØF RECØRDS */
8        WRITE FILE (BB) FROM (PARIT);
9            GØ TØ RD;
10 SØF:  PUT DATA (FIELD);
11       CLØSE FILE (AA), FILE (BB);
12       END RDWT; /* END ØF PRØGRAM */
```

<u>הסבר:</u>

א. בפסוק ההכרזה בשורה 2 ישנה הכרזה של סקלר עם ערך התחלתי אפס, ושל מבנה רשומה. הקבצים מוגדרים בשורה 3. בשורה 4 נרשמה ההתניה למצב של סיום קובץ הקלט, במקרה זה תבוצע הסתעפות לכתובת SOF. בשורה 5 פותחים את שני הקבצים. שים לב שאין צורך להגדיר את התכונות: SEQUENTIAL BUFFERED.

ב. התוכנית מתחילה בהוראת הקריאה והיא נמצאת בלולאה (שורות 9-8-7-6) מבלי להזדקק לבדיקת מצב של סיום הקובץ. מצב זה נבדק אוטומטית בכל פעולת קריאה בהתאם לפסוק ON שקבענו. את רשומת הקובץ קוראים לתוך הרשומה PARIT ומשם גם כותבים אותה.

ג. בסיום העיבוד של הקלט מדפיסים את המונה FIELD (שורה 10) ואח"כ סוגרים את הקבצים (שורה 11).

עיבוד הרשומה בתוך המאגר

פעולת קריאה של רשומה אל שטח הרשומה, או פעולת כתיבה של רשומה מתוך שטח הרשומה גוררת פעולת העברה אשר אפשר, וברוב המקרים גם רצוי, למנוע אותה. כלומר, אפשר לעבד את הרשומה בהיותה במאגר הקלט, או לערוך אותה ישירות אל תוך מאגר הפלט. מכאן משתמע שהשיטה מתאימה לקבצים המוגדרים: RECORD, SEQL, BUF. כדי לבצע את העיבוד בצורה זו נדרשת ידיעה על המקום של הרשומה "התורנית" בבלוק הקלט או בלוק הפלט. משיגים זאת ע"י שימוש במצביע (pointer). דרושה הכרזה של משתנה כמצביע לרשומה במאגר, ובנוסף - לציין את השימוש בו בהוראות המתאימות.

כדי לפעול עם מצביע נדרשות שתי פעולות הכרזה: האחת - הכרזת המצביע:

```
DECLARE (משתנה) PØINTER;
```

"משתנה" הוא מסוג FIXED BINARY.

```
DECLARE M PØINTER;
```

אם לא מגדירים את המצביע הוא מוגדר כברירת מחדל כאשר המהדיר מזהה את השימוש בו. ההכרזה השניה הינה הכרזת הקשר של המצביע אל הרשומה שאותה הוא משרת. ההכרזה נעשית בצמוד לשם הרשומה, למבנה הראשי - ברמה 1, עם המציין: (מצביע) BASED.

```
DECLARE 1 SAPAK BASED (M),
              2 SHEM CHAR (20),
              2 KTØVET CHAR (30);
```

M הוא משתנה-מצביע הנקרא based variable ותפקידו להצביע על המקום המתאים של הרשומה התורנית בבלוק.

הוראת הקריאה היא:

```
READ FILE (שם הקובץ) SET (מצביע);
READ FILE (MELAY) SET (M);     כמו
```

"המצביע" הוא המשתנה המצביע על מקום הרשומה במאגר (במקום לבצע את העברת הנתונים אל שטח הרשומה). המתכנת מניח שמבנה הרשומה כאילו מוצב בתוך הבלוק החל מהכתובת המוגדרת על ידי "המצביע".

הוראת הכתיבה למקרה זה אינה WRITE אלא LOCATE. משמעותה - איתור מקום הרשומה התורנית בבלוק הפלט בכדי לאפשר עריכת הנתונים "בתוכה". הוראת LOCATE בודקת אם עדיין יש מקום בבלוק. אם הבלוק "מלא" - הוא יכתב בקובץ והמצביע יורה על תחילת בלוק חדש.

LØCATE (שם רשומה) SET (מצביע) FILE (שם הקובץ);

"שם רשומה" הינו המבנה המוכרז בפסוק DECLARE.

"מצביע" הוא המשתנה הקשור באותה רשומה.

לדוגמה:

```
LØCATE SAPAK SET (M) FILE (HESHBØN);
```

כאשר המתכנת בוחר לעבד את הרשומה בתוך המאגר, מחובתו להעביר את השדות הרצויים לו מהמאגר או אל המאגר ע"י פסוקי השמה מתאימים. בעיקר אמורים הדברים לגבי קובץ פלט, שם הוא חייב לאפס או להציב תוי מרווח בשדות שלא מועברים אליהם נתונים אחרים.

תרגיל

נתון קובץ לקוחות המפעל. יש להכין קובץ פלט אשר יכיל את רשומות הספקים החייבים מעל -.10,000 $. יש לתכנת הוראות קלט/פלט לעבודה בתוך המאגרים. מבנה הרשומה נתון בגוף התוכנית.

```
1  CUST:  PRØCEDURE ØPTIØNS (MAIN);
2    DCL  MASTER FILE RECØRD BUFFERED,
3         HØVØT FILE RECØRD ØUTPUT BUFFERED;
4    DCL  1  CUSTREC BASED (M),
5            2 ACCTNØ   FIXED (6)
6            2 NAME     CHAR (25)
7            2 BALANCE FLØAT;
8    DCL  1  PELET BASED (K) LIKE CUSTREC;
9    ØN ENDFILE (MASTER) GØ TØ FIN;
10 START: READ FILE (MASTER) SET (M);
11               IF CUSTREC.BALANCE > 10000 THEN DØ;
12                  LØCATE PELET SET (K) FILE (HØVØT);
13                  PELET = CUSTREC; END;
14               GØ TØ START;
15 FIN:   CLØSE FILE (HØVØT);
16          END CUST;
```

הסבר:

- מכיון שהעיבוד נעשה במאגרים, הופעלו בהוראות המציינים LOCATE ,SET ,BASED.
- הגדרת הקבצים בשורות 3-2 נעשתה בהנחה של ניצול ברירת המחדל.
- בשורה 8 מוגדרת רשומה PELET של קובץ החובות במבנה זהה (LIKE) לרשומת קובץ האב. הציון LIKE איננו מעתיק את תכונת הרשומה BASED. לכל רשומה מצביע נפרד.
- התוכנית תפתח ותסגור אוטומטית את הקבצים שאין עבורם הוראות מפורשות.
- עיקר התוכנית הינה הלולאה בשורות 10 עד 14.

פרק 9: מבנה תוכנית

התוכנית מורכבת מאוספים של פסוקי הגדרה, פסוקי ביצוע ופסוקי בקרה. אוסף פסוקים הכולל יחידה לוגית אחת מכונה בלוק (block). קיימים ב-PL/I בלוקים משני סוגים: PROCEDURE blocks, BEGIN blocks.

PROCEDURE block

PROCEDURE - הינו נוהל המכיל אוסף של פסוקים המכוונים לבצע מטרה מוגדרת. בכל תוכנית קיים נוהל מרכזי אחד לפחות, ולכן הגדרת הנוהל המרכזי והראשון של התוכנית הינה:

PRØCEDURE ØPTIØNS (MAIN);

בהגדרה זו השתמשנו בדוגמאות ובתרגילים שהוצגו עד עתה.

אפשרויות נוספות לנוהלים הם נוהלי-שגרות ונוהלי-פונקציות אשר נעמוד בהמשך על התכונות והאפשרויות העיקריות לכתיבתם. הקורא המעינין להרחיב ידיעותיו יפנה לספרות המומלצת.

שגרות

אפשר לכלול בתוכנית נוהלים נוספים אשר אפשר לכתוב ברצף עם הנוהל הראשי או בנפרד. כאשר הם נכתבים בנפרד הם יכולים לעבור הידור ונפוי נפרד, ורק בשלב הסופי הם מצטרפים לנוהל הראשי ע"י פעולת LINK EDIT של מערכת ההפעלה.

נוהלים משניים הינם בדרך כלל שגרות המכוונות לבצע פעולה מסויימת מוגדרת, החוזרת על עצמה פעמים אחדות בתוכנית, או שהן חוזרות בתוכניות אחדות ועל כן נכתבות פעם אחת בלבד.

כל נוהל מהווה יחידת תוכנית סגורה ועל כן אפשר לחזור בכל נוהל על אותם השמות של המשתנים ללא חשש לרב-משמעויות.

הפרמטר של הנוהל הראשי הוא הציון MAIN. בנוהלים משניים ישנם גם כן פרמטרים, הכוללים את רשימת המשתנים המועברים בינם לבין נוהלים אחרים. הפרמטרים, או המש-תנים, צריכים להיות שוים במבנה ובסדר, אבל שם הפרמטר עצמו אינו חשוב.

פנייה של נוהל אחד אל נוהל אחר נעשית ע"י ההוראה CALL, בעלת המבנה הבא:

CALL (שם הנוהל) (רשימת פרמטרים);

- "שם הנוהל" הינו התוית שלפני פסוק PROCEDURE.
- "רשימת הפרמטרים" ניתנת בסוגריים וכוללת את שמות המשתנים. אפשר לרשום גם את התכונות של המשתנה המוחזר. רשימה זו אינה קיימת, אם אינה דרושה.

חזרה מנוהל הינה ע"י ההוראה RETURN. השימוש בהוראה זו אינו חובה אם היציאה מהשגרה הינה בהוראה END.

תרשים להצגת הסתעפות בין נוהלים:

נוהל ראשי	נוהל משני
PGM: PRØCEDURE ØPTIØNS (MAIN)	XXXX: PRØCEDURE (A,B,C)
פסוקי התוכנית	פסוקי התוכנית
------------	------------
------------	------------
CALL XXXX (X,Y,Z);	------------

------------	RETURN;
END;	END;

לדוגמה:

הנוהל המרכזי:

```
START: PRØCEDURE ØPTIØNS (MAIN);
       DECLARE (A,B,C) FIXED (5,2),
               M CHAR (10);
       -------------
```

קריאה לשגרה:

```
       CALL VALUE (M,A,C);
       -------------
       END START;
```

השגרה:

```
VALUE: PRØCEDURE (K,X,Y);
       DECLARE K CHAR (10),
              (X,Y) FIXED (5,2);
       -------------
```

סיום השגרה:

```
       RETURN;
       END VALUE;
```

פונקציות

לעתים נוח להשתמש בשם נוהל בתוך ביטוי אריתמטי, כלומר שלא בעזרת ההוראה CALL. נוהל זה מוכר כנוהל-פונקציה מכיון שהשימוש בו דומה לשימוש ב"פונקציה-בנויה".

דוגמה להפעלת פונקציה MULT למכפלת שני מספרים:

```
TØTAL = MULT (FACTØR, A) + 5.4;
```

מכיון שהנוהל מוזכר בתוך הפסוק ושמו מופיע בו בדומה לכל משתנה אחר, צריך להגדיר את תכונותיו כפי שמקובל עבור משתנים. בנוסף צריך לציין את רשימת המשתנים/פר-מטרים המועברים בינו לבין הנוהל שהוא מוכל בו. מבנה ההוראה:

(שם הנוהל/פונקציה): PRØCEDURE (רשימת משתנים)(תכונות הפונקציה);

לדוגמה:

```
MULT: PRØCEDURE (FACTØR1, FACTØR2) FLØAT (6);
```

חזרה מנוהל-פונקציה חייב להיות בהוראה RETURN המחזירה ערך של משתנה. מקובל להשתמש בהוראה RETURN המלווה בביטוי שערכו מועבר אל הנוהל אשר קרא לפונקציה. אפשר לכתוב את ההוראה גם בתוך משפט, לדוגמה:

```
IF A = B THEN RETURN (1);
```

דוגמה לנוהל-פונקציה לחיבור 2 מספרים:

```
ADD: PRØCEDURE (A,B) FIXED (7,2);
     DECLARE (A,B) FIXED (7,2);
     RETURN (A + B);
     END;
```

בנוהל הראשי נפנה לפונקציה בצורה הבאה:

```
X:   PRØCEDURE ØPTIØNS (MAIN);
     DECLARE (P,Z) FIXED (6,2);
     R = ADD (P,Z);
     PUT LIST (R);
     END;
```

BEGIN block

כדוגמת PROCEDURE block, גם בלוק BEGIN תוחם אוסף של פסוקים בתוכנית. אולם שלא כקודמו שיש לו שימושים מגוונים, מיועד בלוק BEGIN למטרת הגדרת תקי-

פותח של מזהים (שמות). מקובל להשתמש בבלוק זה בפסוקי התנייה ...ON.

המבנה הכללי של הבלוק הוא: (תוית): BEGIN;

(פסוקי התוכנית)

END;

הכניסה לבלוק BEGIN הינה ברצף ההוראות של התוכנית, או ע"י הוראה GØ TØ אם יש לו תוית. התוית אינה הכרחית.

שמות הרשומים בתוך בלוק BEGIN תקיפים בתוכו אולם אינם תקיפים מחוצה לו.

תקיפות הגדרות מזהים

בדוגמאות הקודמות נכתבו נוהלים שונים מסוג PROCEDURE ברצף, אך לא ניתנה דוגמה של נוהל אחד המוכל בנוהל אחר (כדוגמת לולאות DO המוכלות זו בזו). בדוגמאות אלו כל הגדרה תקיפה בתוך הנוהל שלה בלבד. בדומה לא הצגנו דוגמאות של הכלה של בלוקים מסוג BEGIN.

כאשר ישנם נוהלים המוכלים זה בזה יש להקפיד על הכללים לתקיפותם של ההגדרות של המזהים (המשתנים). כללים אלו מפורטים בספרות בנושא Recognition of Names.

פונקציות בנויות

פונקציה מפיקה ערך חדש עבור כל ערך או קבוצת ערכים אשר היא מקבלת. פונקציה המפיקה שורש ריבועי למשל, מקבלת ערך X כלשהו ומפיקה ערך אחר שהוא שורש של X.

הערך המתקבל ע"י הפונקציה נקרא ארגומנט. הפונקציה יכולה לקבל ארגומנט אחד או יותר, אבל תפיק (או תחזיר) ערך אחד בלבד.

דוגמה לפונקציה המקבלת יותר מארגומנט אחד היא פונקציה המחזירה את הערך המירבי מבין אוסף ערכי הארגומנטים המועברים אליה.

צורת הגדרת פונקציות ע"י המתכנת הוסברה בפרק "מבנה התוכנית ב-PL/I". פונקציות בנויות (Built-in functions) הן חלק של השפה ואלה הן בדרך כלל פונקציות בעלות שימוש נרחב.

השימוש בפונקציה בנויה נעשה ע"י קריאה בשמה. הארגומנטים לפונקציה נרשמים בסוגריים הבאים אחרי השם. למשל, SQRT הוא שמה של פונקציה בנויה המחשבת ומחזירה את השורש הריבועי של ערך הארגומנט המועבר אליה. שימוש לדוגמה בפונקציה

זו:

```
M = L + SQRT(RT);
```

משמעות הביטוי: השורש הריבועי של RT יחושב ע"י הפונקציה SQRT, ערך זה יתוסף לערכו של L והתוצאה תושם ב-M. הארגומנט לפונקציה בנויה יכול להיות ביטוי מסוג כלשהו. ובמקרה של SQRT לדוגמה:

```
Y = SQRT (X/5 + 1);
```

פונקציה המקבלת יותר מארגומנט אחד היא הפונקציה MAX. לדוגמה:

```
HIGHEST_VALUE = MAX(P,Q,M-10);
```

לפונקציה MAX יועברו במקרה זה שלושה ארגומנטים. ערכו של P, ערכו של L וערך הביטוי M-10. הפונקציה תחזיר את הערך המירבי מביניהם. ערך זה יושם במשתנה HIGHEST_VALUE.

נהוג לסווג את הפונקציות הבנויות לפי הקטגוריות הבאות: א. אריתמטיות, ב. מתימטיות, ג. טיפול במחרוזות, ד. טיפול במערכים, ה. שונות. הסבר לחלק מן הפונקציות הבנויות ב-PL/I ימצא הקורא בפרקים הבאים.

<u>סקירה על פונקציות נפוצות</u>

השימוש בפונקציות בנויות נפוץ מאד בעבודות מדעיות, ולמעשה לרוב נוהלי החישוב המקובלים ישנן פונקציות מוכנות. נזכיר כאן רק את הפונקציות הטרי-גונומטריות (SIN, COS) ואת הפונקציות הלוגריתמיות.

קבוצה נוספת של פונקציות האופיינית למערכות עיבוד נתונים מינהלי הן הפונקציות לטיפול במערכי נתונים והפונקציות לטיפול במחרוזות נתונים.

פונקציות לטיפול במערכים:

SUM - סכום כל איברי מערך.
PROD - מכפלת איברי מערך.
ALL, ANY - סריקת סביות במערך לעומת "מסכה" נתונה.
DIM, HBOUND, LBOUND - הצגת תחומי מימד של מערך.

פונקציות לטיפול במחרוזות:

LENGTH - אורך מחרוזת נתונה.
SUBSTR - שליפת חלק ממחרוזת לפי דרישה.
INDEX - סריקת תוים או סביות במחרוזת לעומת "מסכה" נתונה.
TRANSLATE - החלפת תוים או סביות במחרוזת נתונה לפי קביעה מראש. בפונקציה זו אפשר להשתמש לתרגום קודים למשל.
BOOL - ביצוע פעולה בוליאנית רצונית על מחרוזות.

לכל הפונקציות שהוזכרו ישנה רשימת ארגומנטים (סקלרים או מערכים) - אך אלו לא הוזכרו כאן.

שתי פונקציות נוספות נפוצות הן:

TIME - מחרוזת תוים המציגה את הזמן הנוכחי (לפי שעון המחשב) עד דיוק של אלפיות שניה.

DATE - מחרוזת תוים המציגה את התאריך (לפי הנתון במחשב) לפי: יום, חודש, שנה.

התערבויות במהלך ביצוע תוכנית

התערבויות (interruptions) הינן אירועים בלתי מתוכננים אשר יכולים לחול במהלך הביצוע של התוכנית.

תוכנית שתוכנתה ועברה הידור עדיין אינה מבטיחה פעולה תקינה. כדי להבטיח זאת מבצעים נפוי המכוון לנפות את השגיאות העלולות להשאר, אולם למרות זאת יכולים להיות מצבים שאי אפשר לאתרם והם מותנים בנתונים המוזרמים בתוכנית. עד כאן - הצד השלילי, אך ישנם גם מצבים ידועים ומוכרים אשר המתכנת מצפה להם. אולם, בגלל האקראיות שבהופעתם עליו להכין את האפשרות לטפל בהם מחוץ למסלול הרגיל של התוכנית.

נזכיר כאן את האירועים הנפוצים ביותר בתוכניות:

OVERFLOW / UNDERFLOW - גלישה של ערך מספרי בשיטת הנקודה הצפה מעל ומתחת לגבולות השדה.

FIXEDOVERFLOW - גלישה של ערך מספרי בשיטת הנקודה הקבועה מעל קיבולת השדה.

ZERODIVIDE - נסיון לחלק ערך מספרי באפס.

CONVERSION - הסבה לא נכונה של נתונים מספריים, או גלישה מתחום קיבולת השדה.

SUBSCRIPTRANGE - מציין של מערך גולש מתחומו.

ENDFILE(שם קובץ) - מצב של סיום קובץ קלט.

ENDPAGE(שם קובץ) - סיום דף ההדפסה בקובץ פלט למדפסת. נדרשת הוראה למעבר לדף חדש.

כדי להשתמש באפשרות השליטה בהתערבות צריך לרשום פסוק התניית התערבות בצורה הבאה:

ØN (אירוע)(יחידת תכנות);

- "יחידת תכנות" היא פסוק תקף ב-PL/I או בלוק BEGIN (אך אסור קבוצת DO).
בתוך בלוק BEGIN אפשר לכלול הוראות כלשהן ללא מגבלה.

רצוי לרשום פסוק זה בקבוצת ההגדרות שבתחילת התוכנית. לדוגמה:

```
a)  ØN ENDFILE GØ TØ SØF;
b)  ØN ENDFILE BEGIN: CLØSE (SERET);
                      STØP;
               END;
c)  ØN ZERØDIVIDE BEGIN: MØVE ZERØ TØ FIELD;
                         GØ TØ NEXT;
                  END;
```

<u>תרגיל סיכום</u>

<u>נתון</u>: שני קבצים של מחירוני פריטים: מחירון בלירות ומחירון בדולרים. לכל פריט יש רשומת מחיר בלירות ורק לחלקם - מחיר בדולרים. יש לייצר מחירון אחד לכל הפריטים וגם להדפיסו.

<u>התוכנית</u>: בדף הבא.

<u>הסבר לתוכנית</u>: קבצים רציפים מוגדרים במבנה רשומות בבלוקים באורך קבוע. העיבוד מבוצע בשטח הרשומה. רשומות מחיר בלירות יקראו אל רשומת CAT, אשר תהיה גם רשומת קובץ הפלט לאחר עדכונה. רשומת מחיר בלירות היא ITEM.
התניית סוף קובץ מחירון בלירות מבצעת הסתעפות לסוף התוכנית. סיום המחירון בדולרים לפני סיום המחירון בלירות הינו מצב שגיאה אשר ידווח, ותבוצע הסתעפות לסוף התוכנית.
קובץ הפלט להדפסה מוגדר למדפסת. רשומת הקובץ PRNTCAT הינה במבנה מותאם לעריכת פלט הדפסה. תו ראשון הוא תו בקרה לקידום שורות במדפסת. כאן ננצל את ברירת המחדל שלו - קידום שורה אחת.
שני משתני תויות מיועדים להעברת פיקוח התוכנית לתוית הרצויה: כאן - קריאת רשומות בקבצים כפי הצורך.
לאחר קריאת הרשומות נערכת השואה ועדכון רשומת הפלט (CAT): מוסיפים מחיר בדולרים או מאפסים את השדה כאשר אין מחיר כזה. כאשר לפריט אין רשומה בלירות מפסיקים העיבוד ועוברים לסיום התוכנית, ללא דיווח שגיאה (רצוי לדווח). לאחר סיום העדכון כותבים את הרשומה בקובץ הפלט, מעבירים אותה לרשומת העריכה להדפסה וכותבים למדפסת.
בסיום התוכנית - סגירת הקבצים.

PL/I STATEMENT

```
CATALOG: PROCEDURE OPTIONS (MAIN); /* CREATE ONE CATALOG */

    DCL (ILCAT,USCAT) FILE RECORD INPUT SEQL           /* INP FILE */
        ENV ( FB RECSIZE(36) BLKSIZE (1440) );

    DCL (STDCAT) FILE RECORD OUTPUT                    /* OUT FILE */
        ENV (FB RECSIZE(41) BLKSIZE (1640) );

    DCL 1 ITEM,    2 PRNO    CHAR (6),                 /* USCAT RECORD */
                   2 DESC    CHAR (25),
                   2 PRICE   FIXED (8, 2),

    DCL 1 CAT,     2 PRNO    CHAR (6),                 /* ISCAT RECORD */
                   2 DESC    CHAR (25),                /* AND OUTPUT */
                   2 ILPR    FIXED (8,2),
                   2 USPR    FIXED (8,2);

    ON ENDFILE (ILCAT) /* END OF ILCAT FILE */
                   GO TO FINISH ;                      /* CORRECT EXIT */

    ON ENDFILE (USCAT) /* END OF USCAT FILE--SHOULD NOT
                   ACCUR BEFOR ILCAT */
        BEGIN; PRNTCAT = '* ERROR - USCAT TOO LONG *' ;
               WRITE FILE (REPORT) FROM(PRNTCAT); GO TO FINISH;
        END;
  DCL (REPORT) FILE RECORD OUTPUT SEQL
        ENV ( F BLKSIZE(100) CTLASA ) ;  /* CTLASA-FOR LINE SPACING */

  DCL  1 PRNTCAT,  2 CTL     CHAR(5),          /* PRINTED OUTPUT */
                   2 PRNO    PIC '99/9999',          /* EDITED */
                   2 FILLER  CHAR (2) INIT (' '),    /* FIELDS */
                   2 DESC    CHAR (27),
                   2 ILPR    PIC '99,999V.99',
                   2 FILL    CHAR (2) INIT (' '),
                   2 USPR    PIC '99,999V.99';
    DCL ( L, BACK ) LABEL;
   OPEN FILE ( ILCAT, USCAT, STDCAT, PRNTCAT );

L1: L=L3;
L2:        READ FILE (ILCAT) INTO (CAT); GO TO L ;
L3:        READ FILE (USCAT) INTO (ITEM); L=L4;

L4: IF CAT.PRNO = ITEM.PRNO THEN DO;
                   USPR = ITEM.PRICE;
                   BACK = L1 ; GO TO WRITE ; END ;

    IF CAT.PRNO < ITEM.PRNO THEN DO;
                   USPR = 0 ; BACK = L2 ; END ; ELSE GO TO FINISH ;

WRITE: WRITE FILE (STDCAT) FROM ( CAT ) ;      /* WRITE OUT FILE */
        PRNTCAT = CAT, BY NAME;                /* MOVE TO PRINT REC. */

        WRITE FILE (REPORT) FROM ( PRNTCAT ); GO TO BACK ;

FINISH: CLOSE ( ILCAT, USCAT, PRNTCAT) FILE ; STOP ;
    END CATALOG;
```

ביבליוגרפיה

בנושאים הנדונים בספר זה ישנה ספרות ענפה של מחברים שונים, ובעיקר - של יצרני המחשבים. הרשימה הבאה כוללת רק חלק מתוך המבחר הרב העומד לרשות הקורא בשפה האנגלית, והיא כוללת חלק רב מהחומר הקיים בשפה העברית.

רשימת הספרות מחולקת לנושאים, כדי להקל על הקורא את העיון בה. הספרים המומלצים סומנו בכוכבית.

א. מבוא לעיבוד נתונים ותרשימי זרימה

**1. עמיהוד, יצחק. המחשב האלקטרוני ועיבוד הנתונים. הוד-עמי, 1968, 1975.

**2. פורסייט, קינן, אורגניק, שטנברג. יסודות תכנות מחשבים. המכון לאמצעי הוראה 1974 (תרגום).

**3. -. יסודות עיבוד הנתונים האלקטרוני בשיטת הלמידה המתוכנתת. הוצאת המכון לפריון העבודה והייצור, 1973.

4. בעל-שם י. יסודות תורת המחשבים. המכון לאמצעי הוראה, 1970.

5. - פרקי קריאה בעיבוד נתונים. בהוצאת המכון לפריון העבודה והייצור, 1967.

6. פינטשוק י. מחשבים אלקטרוניים - כיצד הם פועלים. הוצאת סדן והמכון לפריון העבודה והייצור, 1967.

7. -. מילון למונחי עיבוד נתונים. הוצאת איגוד ישראלי לעיבוד אינפור-מציה, ירושלים, 1975.

**8. אריאל, צבי. מערכת /370 - ספר ללימוד עצמי. י.ב.מ. ישראל בע"מ, SIE-01001, 1975.

**9. -. *Introduction to IBM Data Processing Systems*. Form GC20-1684.

10. Fletcher, Allan (editor). *Computer Science for Management*. Business Publication Ltd., London, 1967.

11. Crawford F.R. *Introduction to Data Processing*. Prentice Hall, 1973.

12. Martino R.L. *Information Management*. McGraw Hill, 1968.

13. -. *An Introduction to Digital Computers*. Control Data Corporation, Form 60241600.

ב. נושאים משולבים

.1** Cashman and Keys. *Data Processing and Computer Programming.* Harper & Row, Publishers, Inc., 1971.

.2** Sanderson Peter C. *Computer Languages.* Butterworth & Co. (Publishers) Ltd., 1970.

.3 Peterson W. Wesley. *Introduction to Programming Languages.* Prentice-Hall Inc., 1974.

.4 Metzger Phillip W. *Managing a Software Project.* Prentice Hall Inc., 1973.

.5** Ledgard Henry F. *Programming Proverbs.* Hyden Book Company Inc., 1975.

6. פלד ישראל. **מבוא למחשבים למדעי החברה - בחינות ופתרונות.** מפעל השכפול, אוניברסיטת תל-אביב, 1974.

ג. תכנות פורטרן

1.** אדיר, יואל. שפת מחשב לתכנות מדעי FORTRAN. הוצאה פרטית, קרית ביאליק, 1973

2. סלומון, דוד שפת תכנות FORTRAN IV. אקדמון, ירושלים, 1971.

.3** McCracken Daniel D. *A Guide to FORTRAN Programming.* John Wiley & Sons Inc., 1961.

.4 Kazmier Leonard J., Philippakis Andreas S. *Fundamentals of EDP and FORTRAN.* McGraw-Hill Inc., 1970

.5 -. *FORTRAN IV for IBM SYSTEM/360 (Self Study).* IBM, 1965.

.6 -. *IBM/360 and IBM/370 FORTRAN IV Language.* IBM, Form GC28-6818.

.7 -. *FORTRAN Instant.* CDC, Form 60189500.

.8 -. *FORTRAN Reference Manual.* CDC, Form 60174900.

.9 -. *Burroughs Medium Systems FORTRAN - Reference Manual.* Burroughs Corp.

.10 -. *Burroughs FORTRAN Self-Learner.* Burroughs Corp.

.11 -. *Burroughs B1700 Systems FORTRAN - Reference Manual.* Burroughs Corp.

.12 -. *NCR Century FORTRAN.* NCR Form 0215.

ד. תכנות קובול

1.** -. ANS COBOL ללימוד עצמי (בעברית). י.ב.מ. ישראל בע"מ, 1973.

2. -. יסודות קובול למערכות בינוניות, תמכין נציגת חברת Burroughs.

3. James A. Saxon and William R. Englander. *ANSI COBOL Programming*. Prentice-Hall Inc., 1972.

4. McCracken D. *A Guide to COBOL Programming*. John Wiley & Sons Inc., 1970.

5. Ferina M.V. *COBOL Simplified*. Prentice Hall Inc., 1968

6. -. *IBM SYSTEM/360 Disk Operating System American National Standard COBOL*. Form GC28-6394.

7. -. *IBM SYSTEM/360 Operating System American National Standard COBOL*. Form GC28-6396.

8. -. *Flow-Charting for COBOL*. Burroughs Corp.

9. -. *Burroughs B1700 Systems COBOL Reference Manual*. Burroughs Corporation.

10. -. *Burroughs B700 Systems COBOL - Reference Manual*. Burroughs Corporation.

11. -. *COBOL General Information Manual*. Control Data Corporation, Form 60327800.

12. -. *COBOL Instant*. CDC, Form 60328400.

13. -. *COBOL Reference Manual*. CDC, Form 60384100.

14. -. *NCR Century COBOL*. NCR Form 0211.

15. -. *NCR Century COBOL Student Text*. NCR.

ח. <u>תכנות PL/I</u>

Bates, Frank and Mary L. Douglas. *Programming Language/One.* .1**
Prentice Hall Inc., 1970.

-. *A PL/I Primer.* IBM Form GC28-6808. .2**

-. *A Guide to PL/I for Commercial Programmers.* .3
IBM, Form GC20-1651.

-. *Preface to PL/I Programming in Scientific Computing.* .4**
IBM, Form GE20-0312.

-. *OS PL/I Checkout and Optimizing Compilers: Language Reference* .5
Manual. IBM, Form SC33-0009.

-. *OS PL/I Optimizer Compiler Programmer's Guide.* .6
IBM, Form SC33-0006.

-. *DOS PL/I Language Reference Manual.* IBM, Form GC33-0005. .7

Weinberg, Gerald M. *PL/I Programming, A Manual of Style.* .8
McGraw-Hill, N.Y., 1970.

Weinberg, Gerald M., Yasukawa, N., and Marcus, R., *Structured* .9
Programming in PL/I. Wiley, N.Y., 1973.

תכנון ותכנות

יישומי מחשבים